

Image Processing

Computational Photography

CSE 273

Lecture 4

Announcements

- Assignment 2 is due Jan 24, 11:59 PM
- Assignment 3 will be released Jan 24
 - Due Jan 31, 11:59 PM

Image sensing pipeline

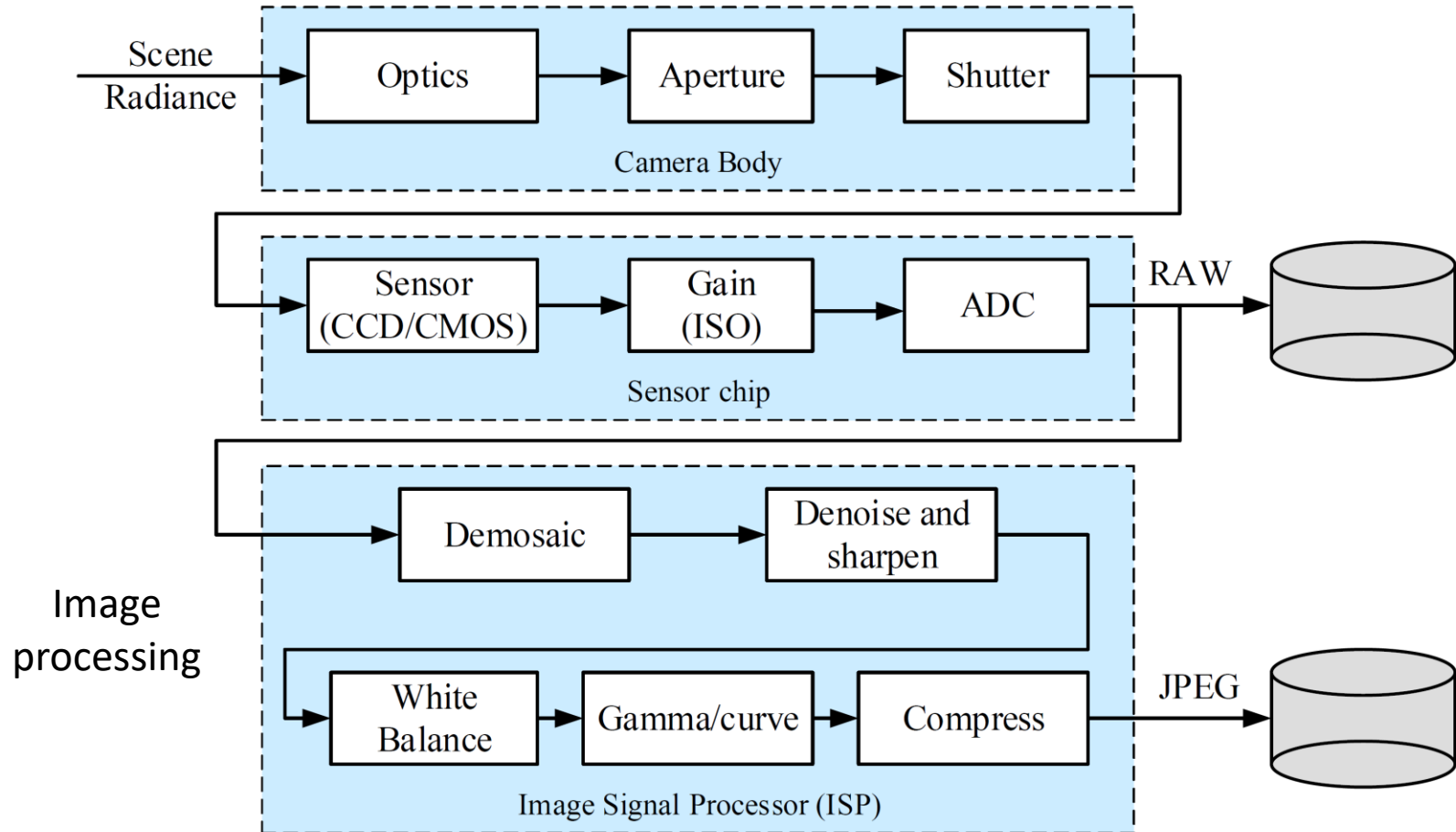
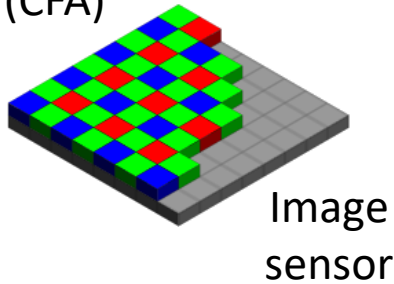


Image processing

- A discipline in which both the input and output of a process are images
 - There are usually other input parameters to the process

Demosaicing

Color filter array (CFA)

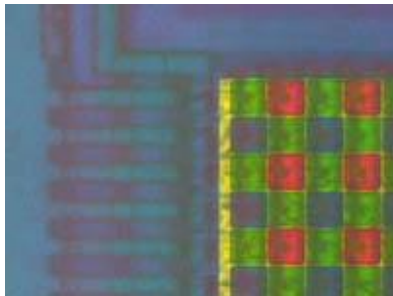


G	R	G	R
B	G	B	G
G	R	G	R
B	G	B	G

CFA (single channel image)

rGb	Rgb	rGb	Rgb
rgB	rGb	rgB	rGb
rGb	Rgb	rGb	Rgb
rgB	rGb	rgB	rGb

RGB (interpolated values in lower case)



Bayer pattern

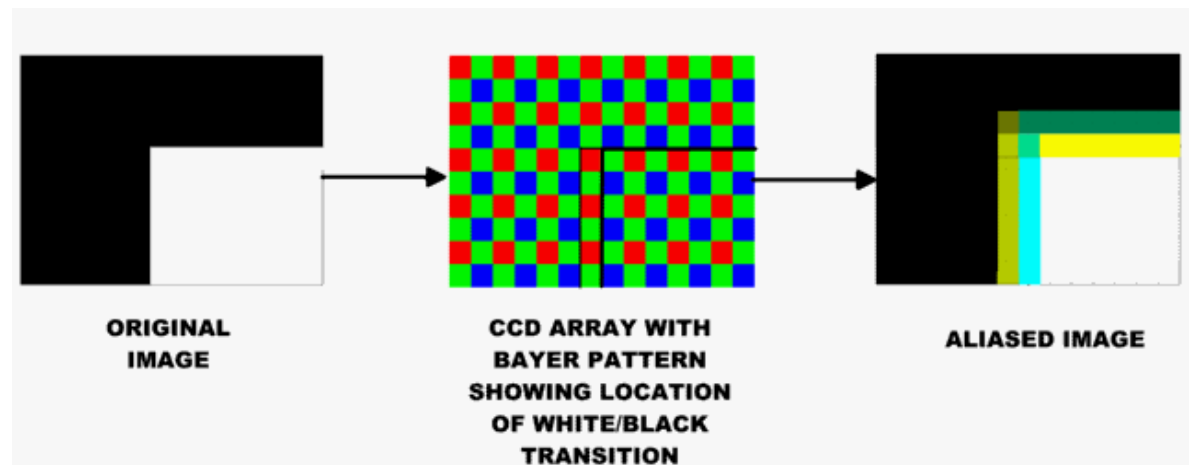


Image processing

- Color spaces
- Gamut mapping
- White balancing and color balancing

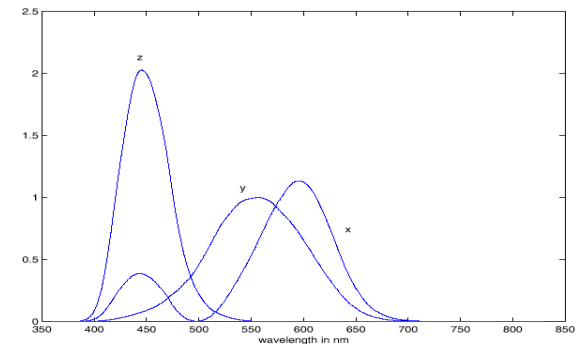
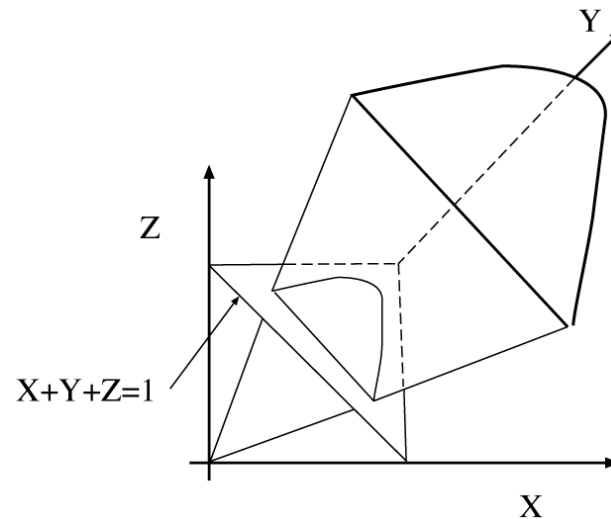
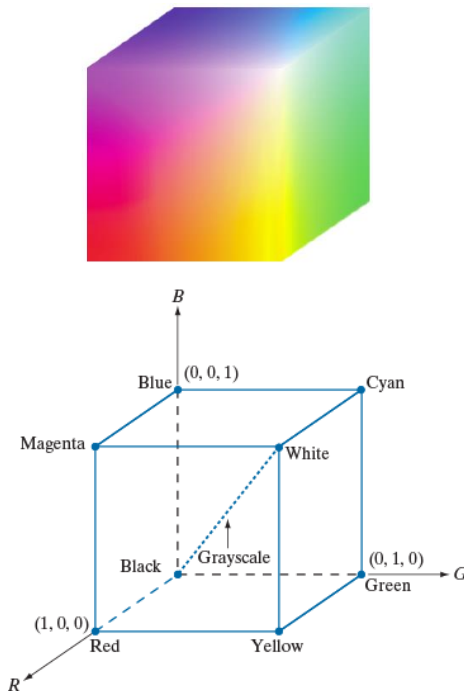


Image processing

- Dehazing
- Denoising
- Deconvolution

Motion blur
and
additive noise

Less noise

Much less noise

Degraded
image

Inverse
filtering

Wiener
filtering

Constrained
least
squares
filtering



2D correlation and convolution

2D correlation (also called cross-correlation)

$$g(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)$$

$$g(x, y) = w(x, y) \star f(x, y), \text{ where } w(x, y) \star f(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)$$

2D convolution

$$g(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x - s, y - t)$$

$$g(x, y) = w(x, y) \star f(x, y), \text{ where } w(x, y) \star f(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x - s, y - t)$$

Equivalent to correlation with w rotated 180 degrees
where

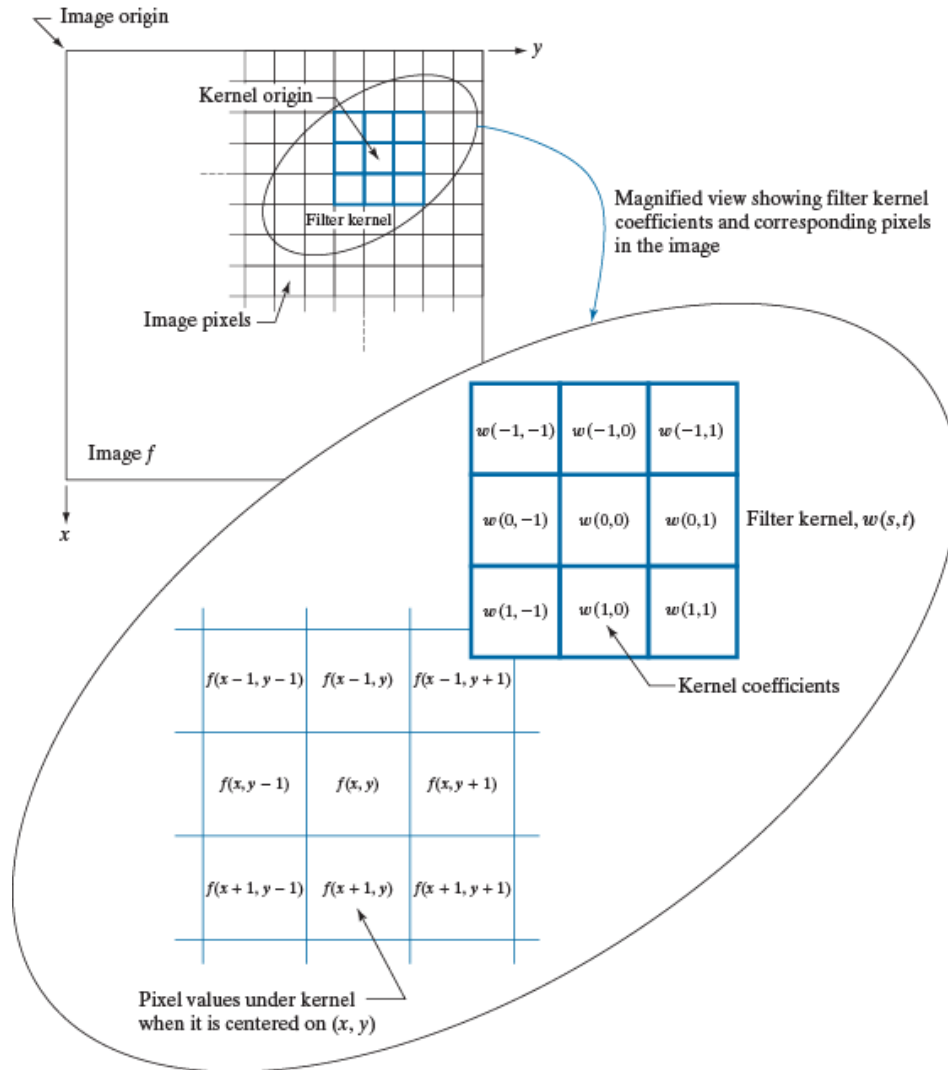
When kernel is symmetric, convolution
and correlation give the same result

$w(s, t)$ is spatial filtering kernel (also called mask, template, and window)

$f(x, y)$ is input image

$g(x, y)$ is output image

2D spatial filtering



$m \times n$ kernel is $(2a + 1) \times (2b + 1)$, where a and b are positive integers, so m and n are odd positive integers

2D correlation

$$w(x, y) \star f(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)$$

2D convolution

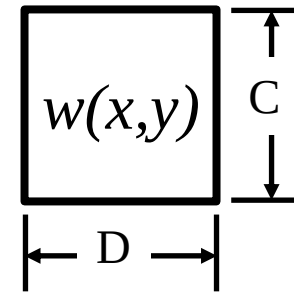
$$w(x, y) \star f(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x - s, y - t)$$

For convolution, kernel is rotated 180 degrees

2D spatial filtering, shape

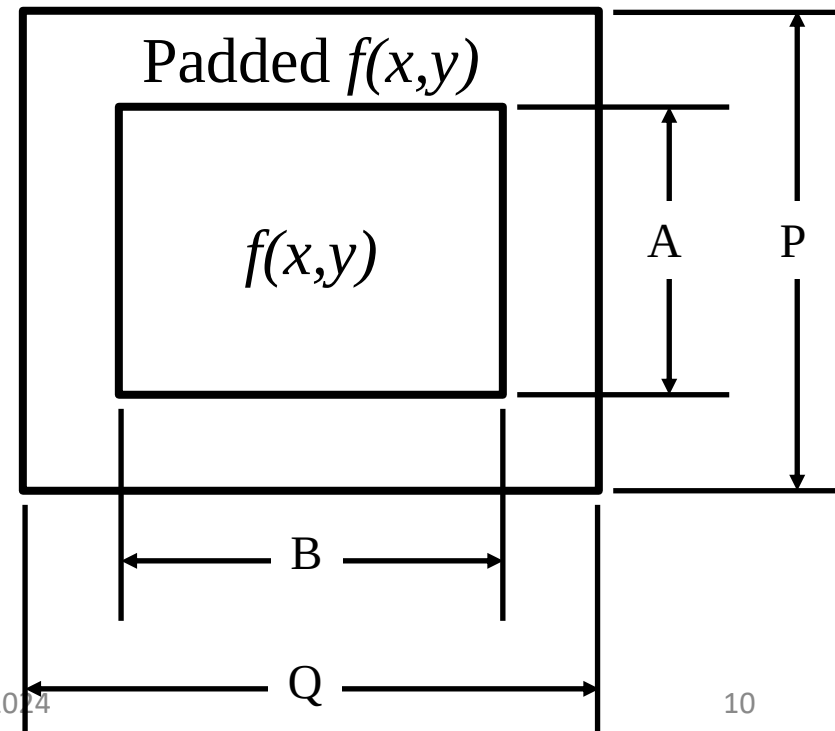
- Full

- $w(x,y)$ and $f(x,y)$ have at least 1 pixel overlap
- $P = A + 2(C - 1)$
- $Q = B + 2(D - 1)$
- Output $g(x,y)$
 - Width is $B + D - 1$, height is $A + C - 1$



- Same

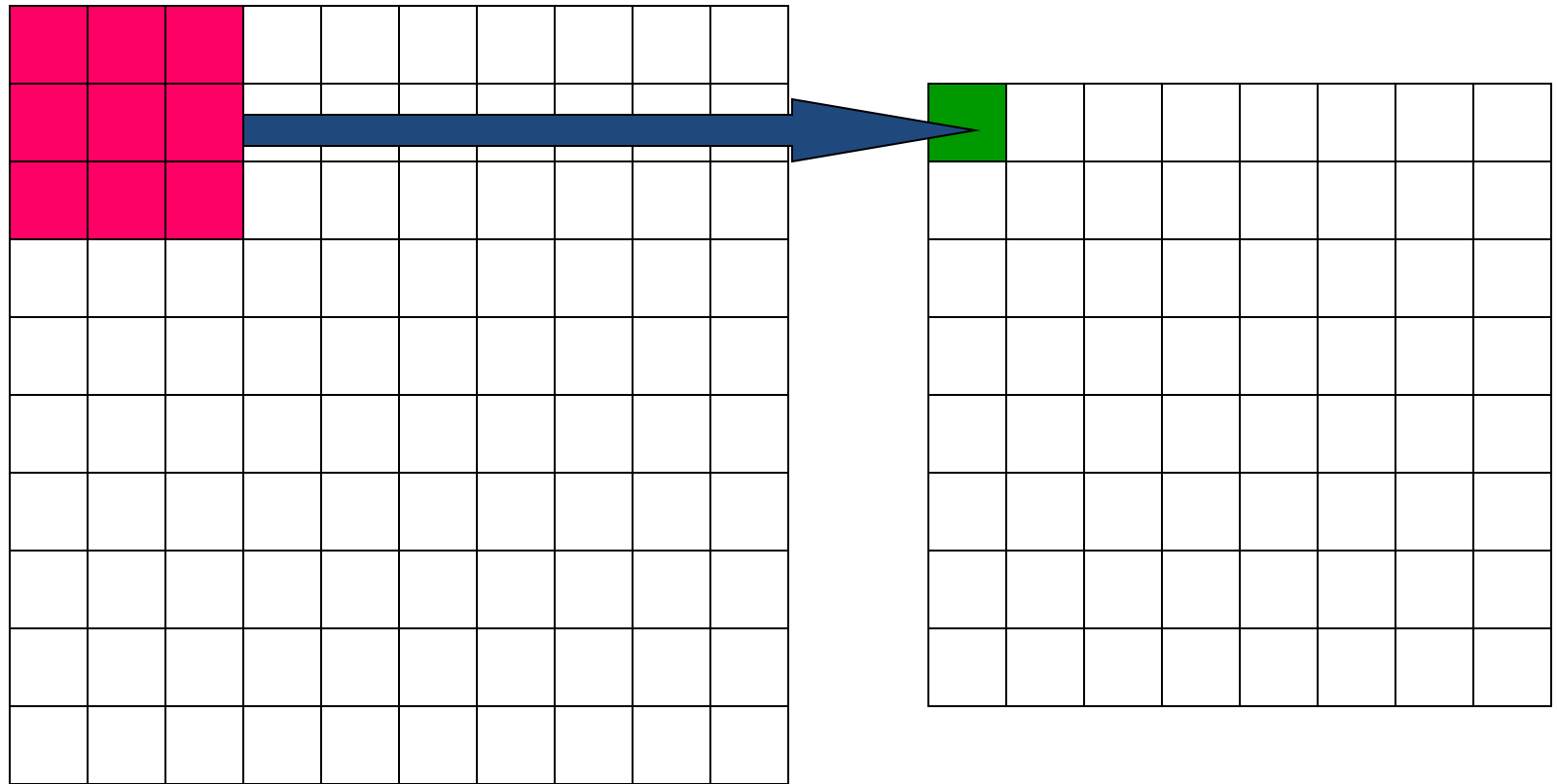
- $P = A + C - 1$
- $Q = B + D - 1$
- Output $g(x,y)$
 - Width is B , height is A



- Valid

- $w(x,y)$ must be completely inside $f(x,y)$
- $P = A$
- $Q = B$
- Output $g(x,y)$
 - Width is $B - D + 1$, height is $A - C + 1$

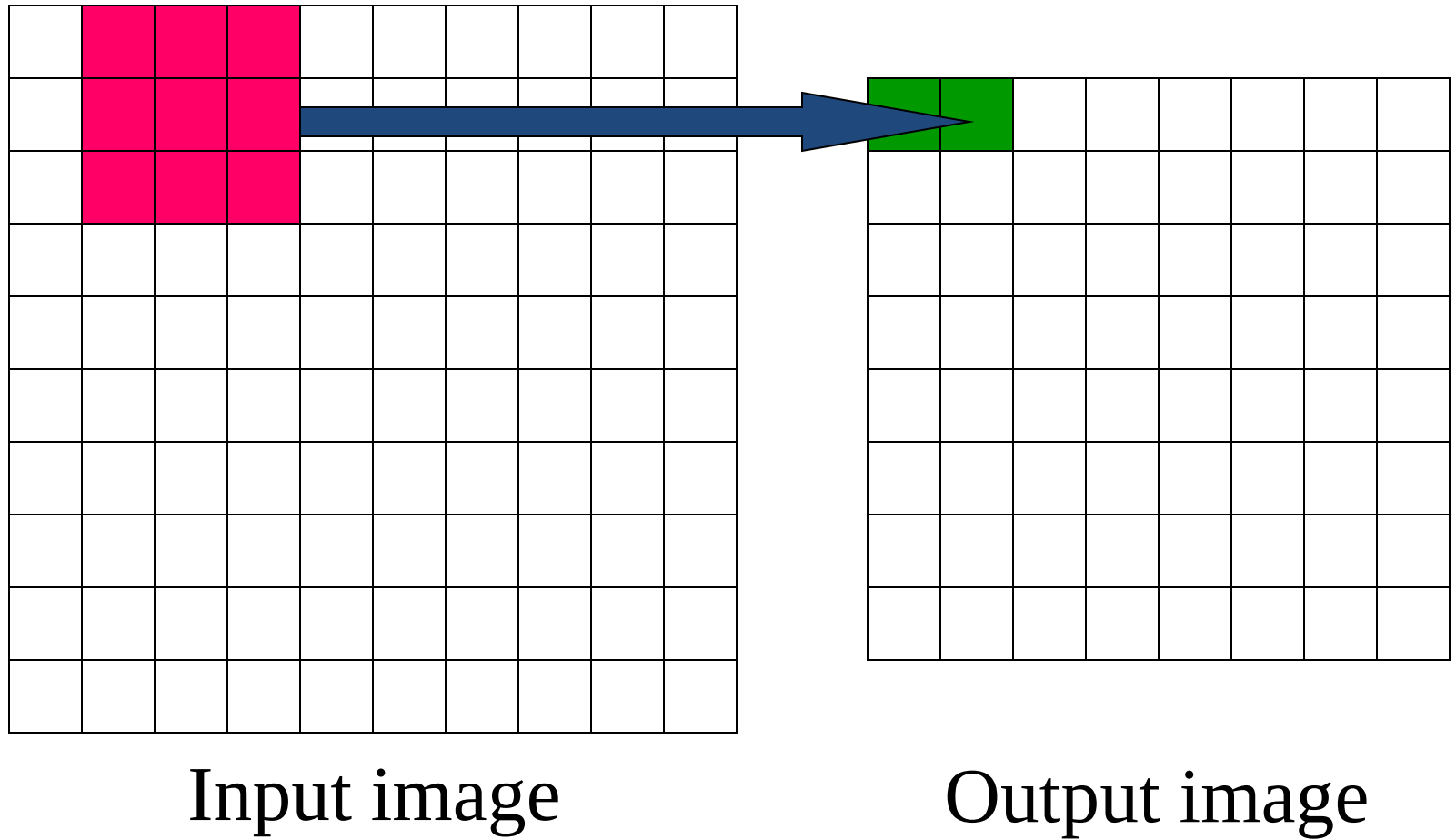
2D spatial filtering, valid shape



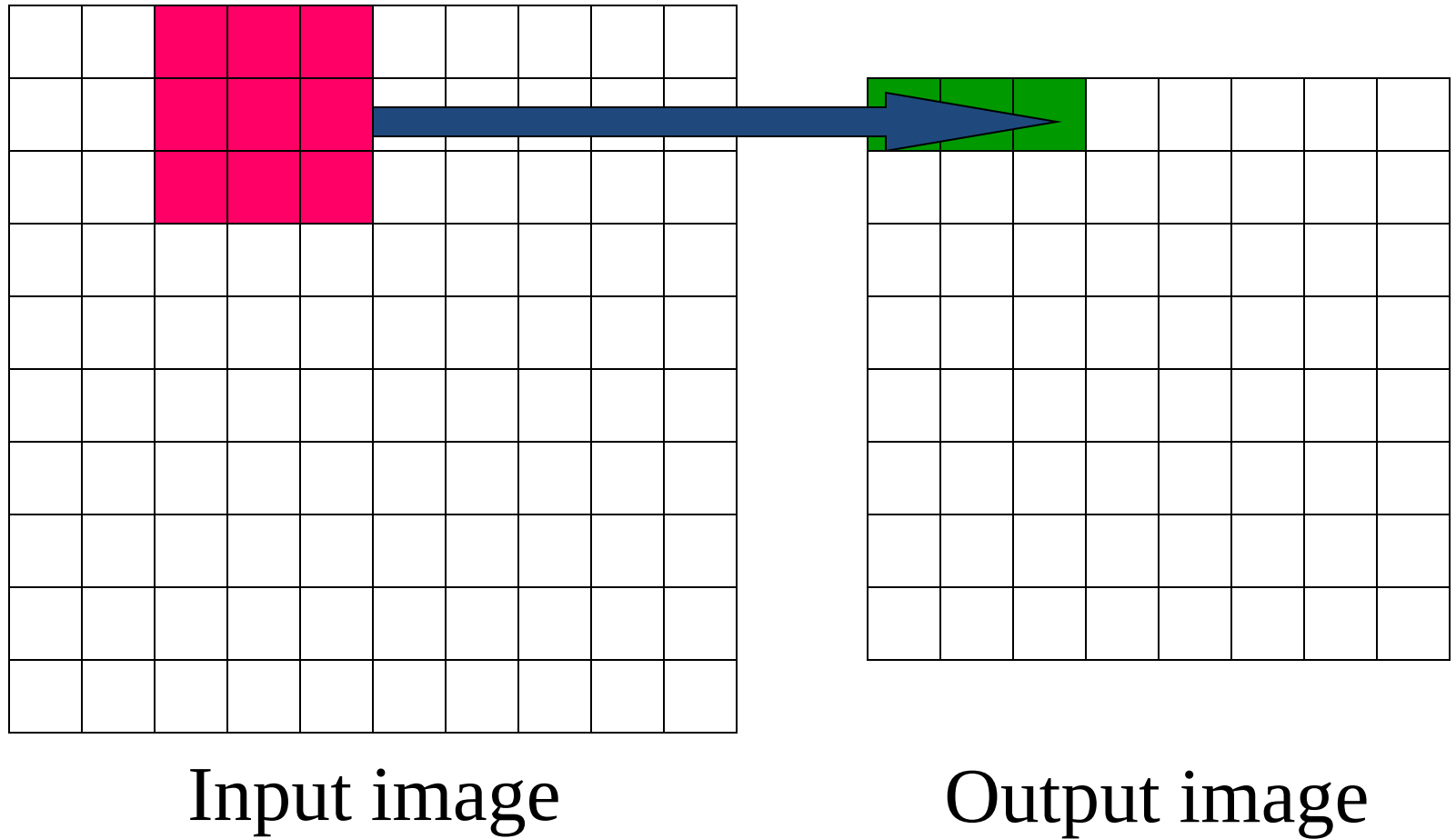
Input image

Output image

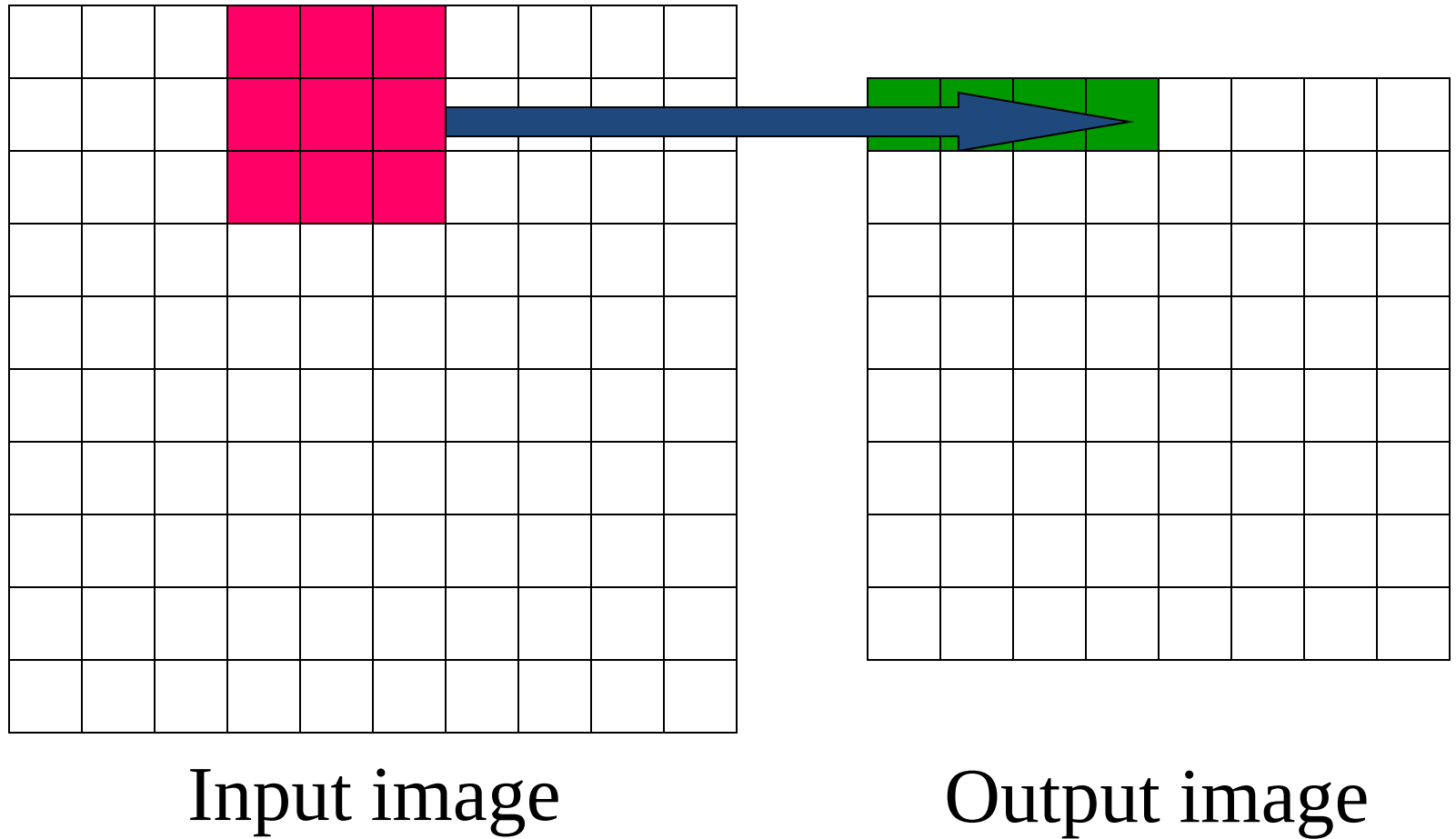
2D spatial filtering, valid shape



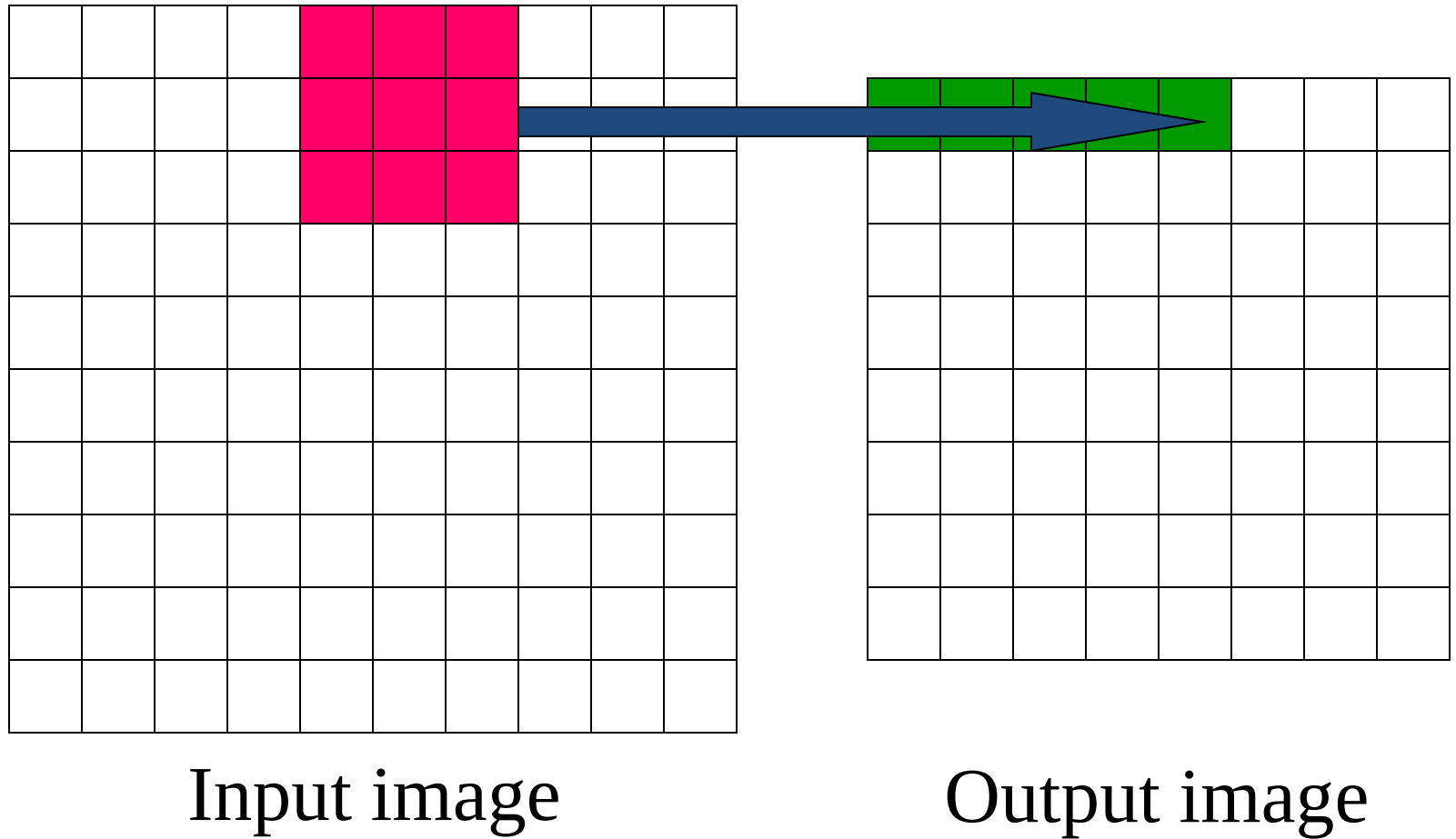
2D spatial filtering, valid shape



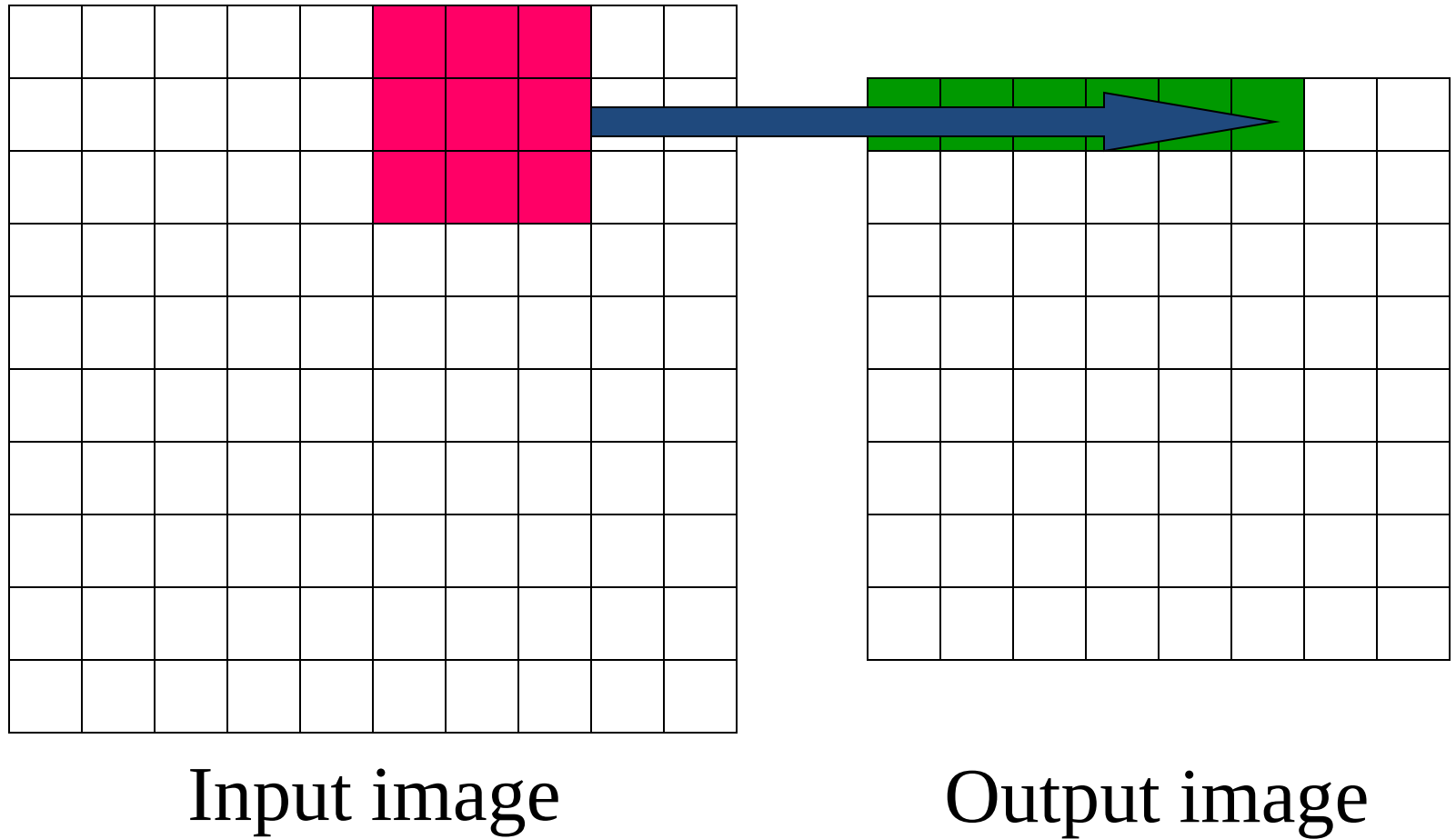
2D spatial filtering, valid shape



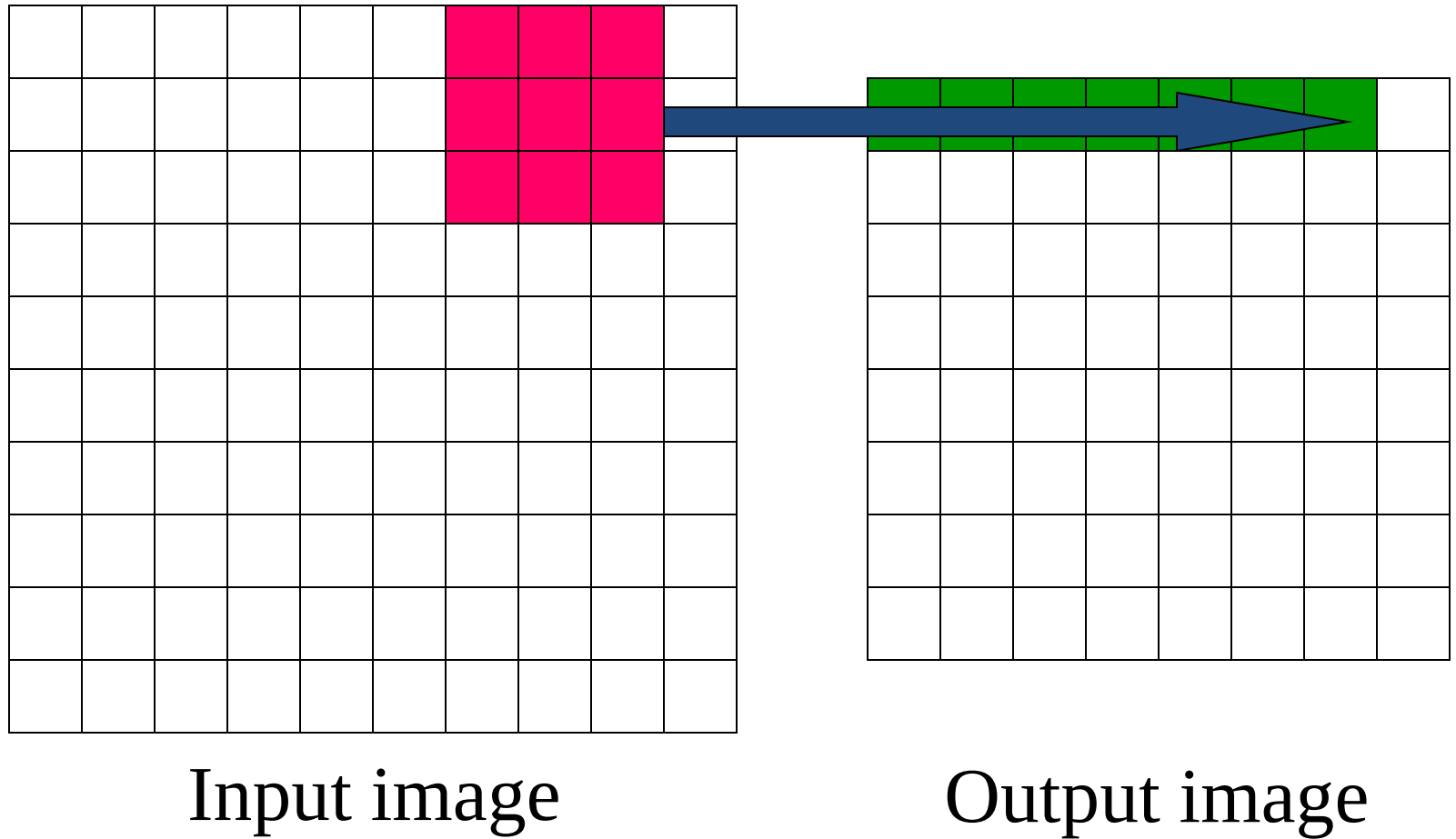
2D spatial filtering, valid shape



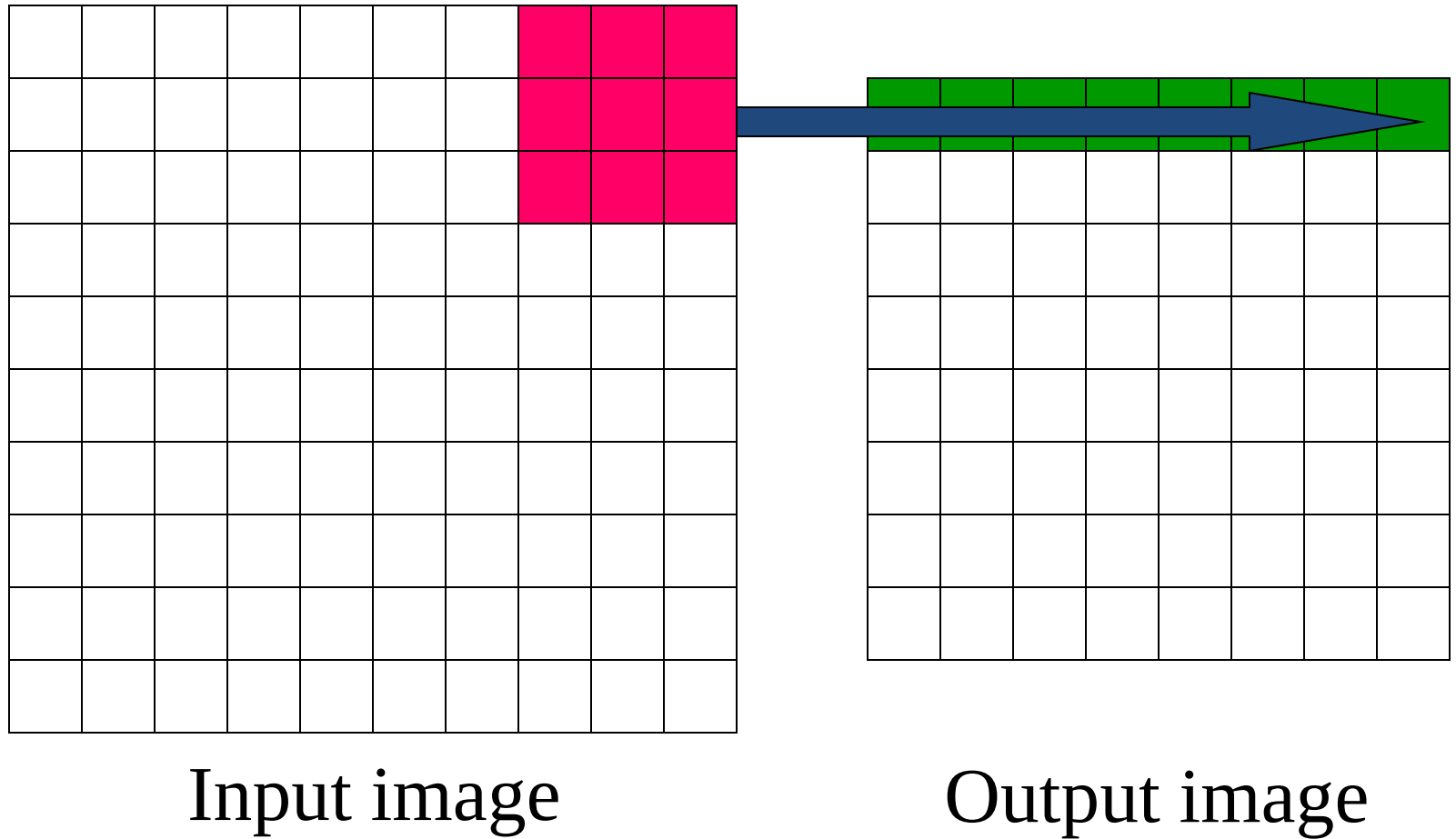
2D spatial filtering, valid shape



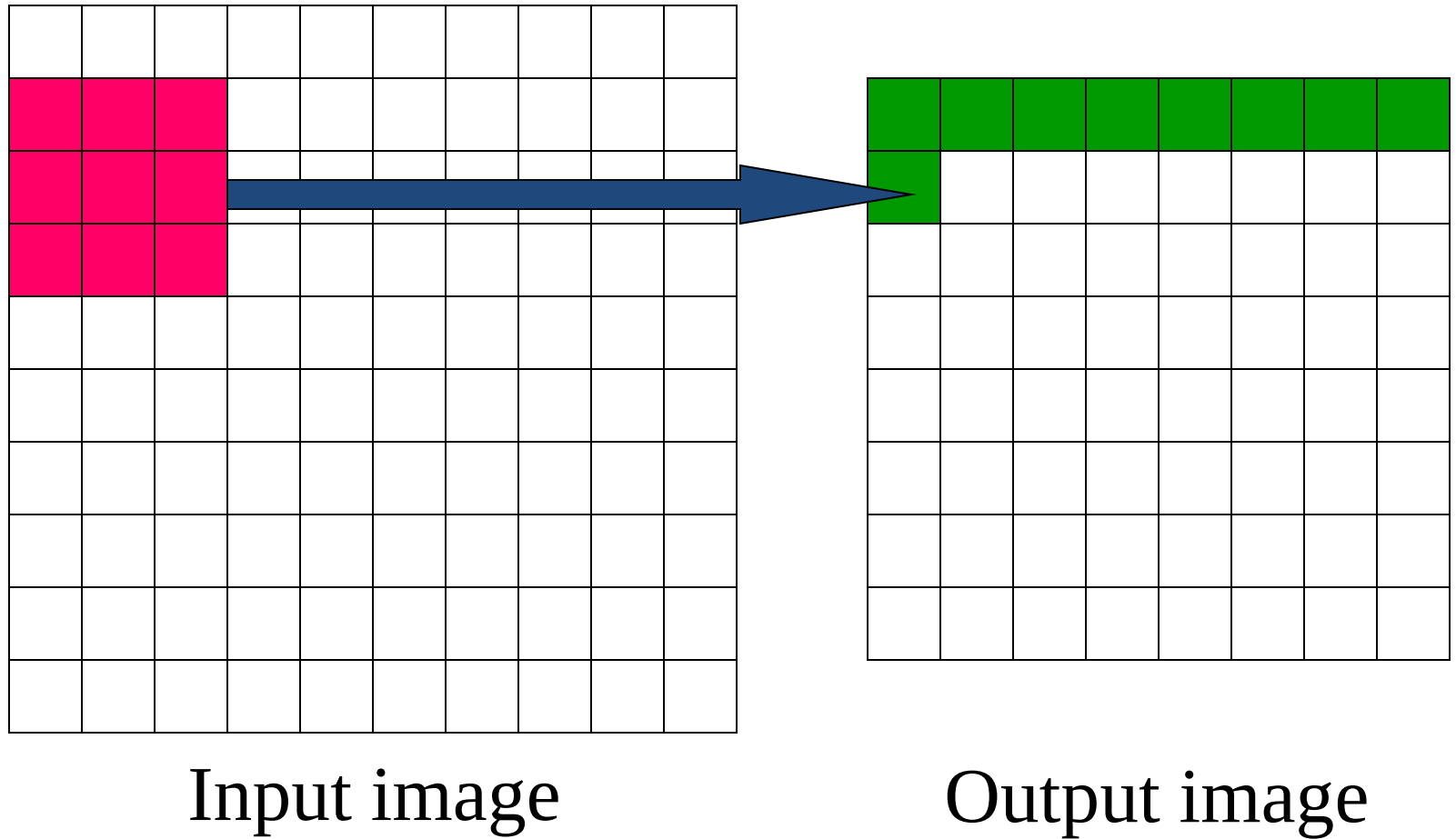
2D spatial filtering, valid shape



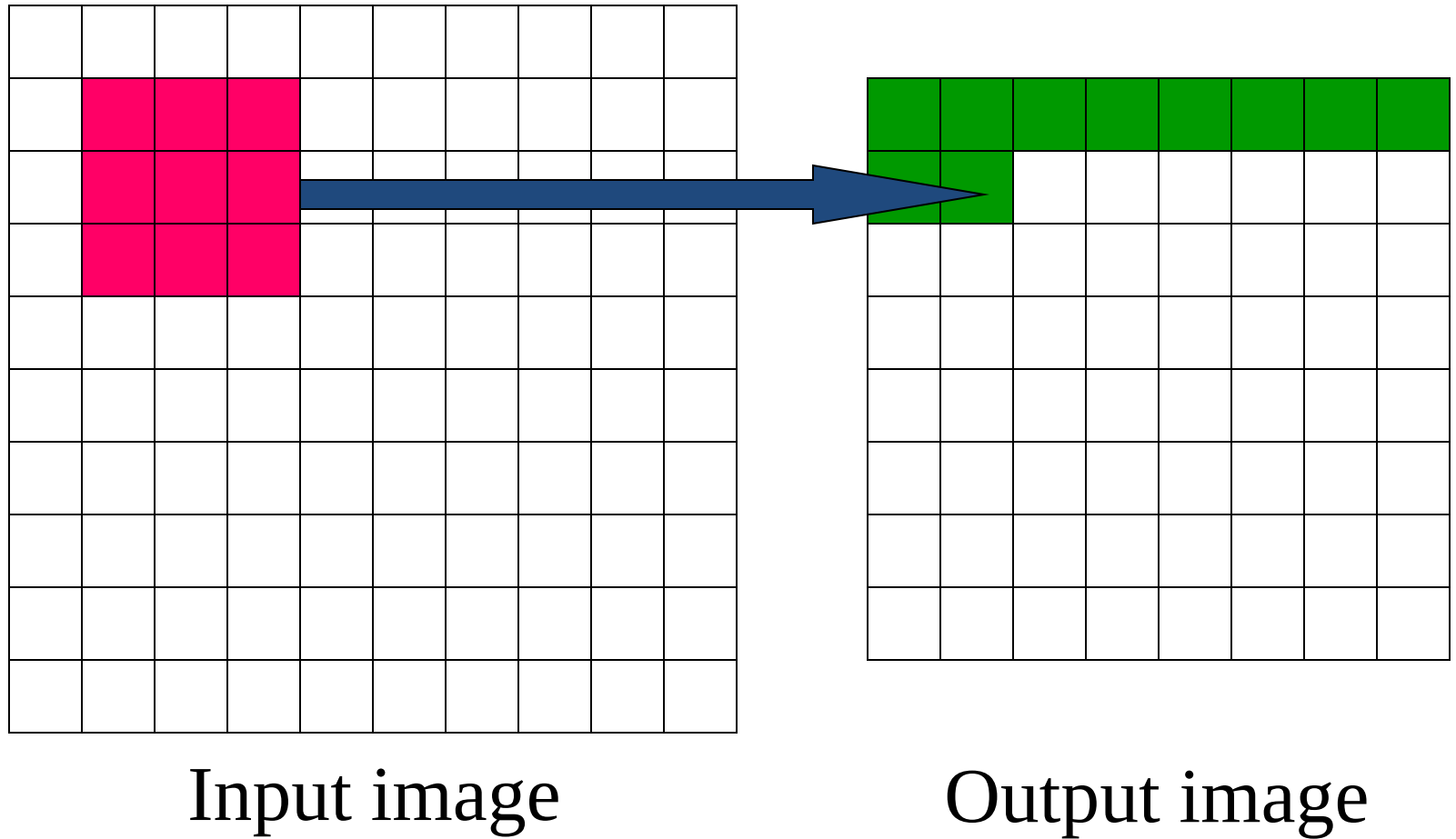
2D spatial filtering, valid shape



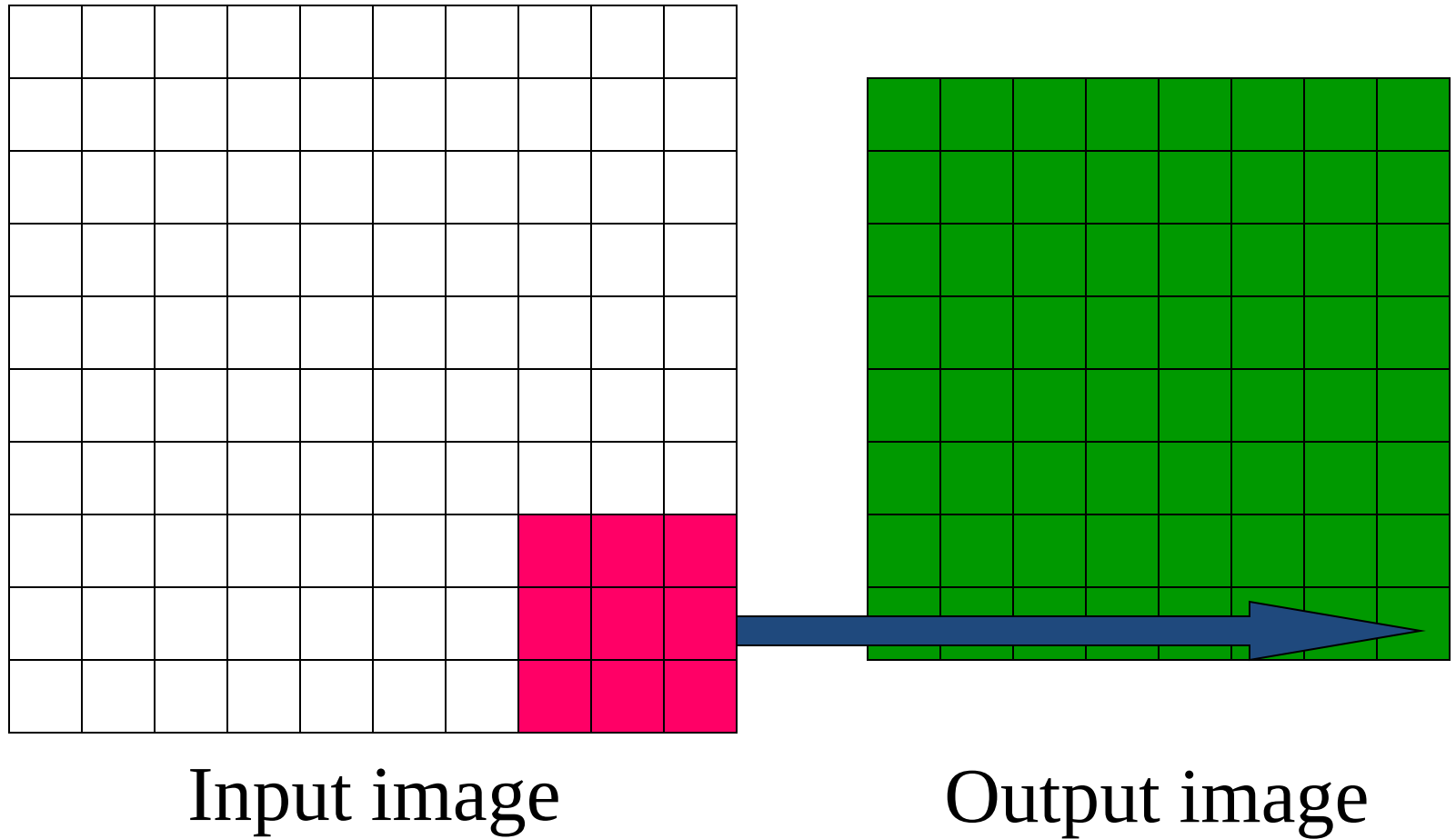
2D spatial filtering, valid shape



2D spatial filtering, valid shape



2D spatial filtering, valid shape



2D correlation and convolution

FIGURE 3.36

Correlation (middle row) and convolution (last row) of a 2-D kernel with an image consisting of a discrete unit impulse. The 0's are shown in gray to simplify visual analysis. Note that correlation and convolution are functions of x and y . As these variable change, they *displace* one function with respect to the other. See the discussion of Eqs. (3-45) and (3-46) regarding full correlation and convolution.

Unit impulse

Origin f

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

(a)

Padded f

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

(b)

For convolution, kernel is rotated 180 degrees

Initial position for w

1	2	3	0	0	0	0
4	5	6	0	0	0	0
7	8	9	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

(c)

Correlation result

0	0	0	0	0	0
0	9	8	7	0	0
0	6	5	4	0	0
0	3	2	1	0	0
0	0	0	0	0	0

(d)

Full correlation result

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	9	8	7	0	0
0	0	6	5	4	0	0
0	0	3	2	1	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

(e)

Rotated w

9	8	7	0	0	0	0
6	5	4	0	0	0	0
3	2	1	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

(f)

Convolution result

0	0	0	0	0	0
0	1	2	3	0	0
0	4	5	6	0	0
0	7	8	9	0	0
0	0	0	0	0	0

(g)

Full convolution result

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	1	2	3	0	0
0	0	4	5	6	0	0
0	0	7	8	9	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

(h)

Correlation and convolution

This is equivalent to applying one filter

TABLE 3.5

Some fundamental properties of convolution and correlation. A dash means that the property does not hold.

Property	Convolution	Correlation
Commutative	$f \star g = g \star f$	—
Associative	$f \star (g \star h) = (f \star g) \star h$	—
Distributive	$f \star (g + h) = (f \star g) + (f \star h)$	$f \star (g + h) = (f \star g) + (f \star h)$

- Convolution is commutative and associative, correlation is not

2D spatial filtering

- Example: low pass (i.e., smoothing) filtering kernels



Smoothing kernels

 $\frac{1}{9} \times$

1	1	1
1	1	1
1	1	1

Average
(box kernel)

 $\frac{1}{4.8976} \times$

0.3679	0.6065	0.3679
0.6065	1.0000	0.6065
0.3679	0.6065	0.3679

Weighted average
(Gaussian kernel)

Common that sum of filter coefficients is 1, which prevents (full) bias
Bias: sum of input pixel values not equal to sum of output pixel values

Smoothing with box kernel

Input
image



3x3

11x11



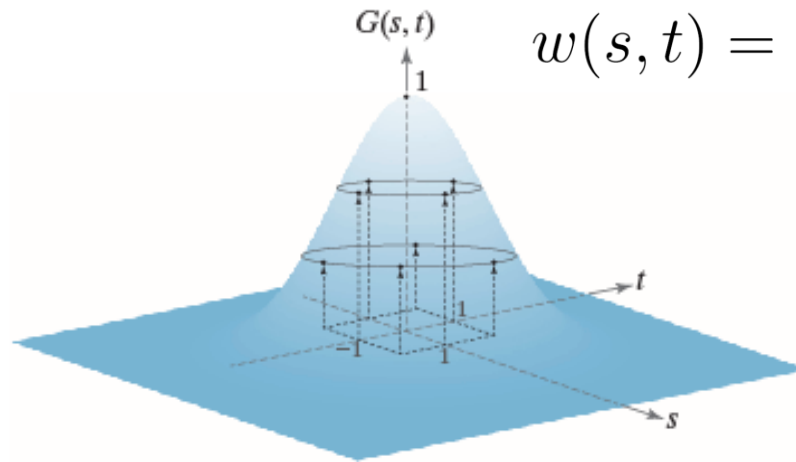
21x21

Smoothing with Gaussian kernel

a b

FIGURE 3.41

(a) Sampling a Gaussian function to obtain a discrete Gaussian kernel. The values shown are for $K = 1$ and $\sigma = 1$. (b) Resulting 3×3 kernel [this is the same as Fig. 3.37(b)].



$$w(s, t) = G(s, t) = K e^{-\frac{s^2 + t^2}{2\sigma^2}}$$

$$\frac{1}{4.8976} \times$$

0.3679	0.6065	0.3679
0.6065	1.0000	0.6065
0.3679	0.6065	0.3679

Standard deviation σ	Percent of total volume under surface
1	39.35
2	86.47
3	98.89

Volume under surface greater than 3σ is negligible

Smoothing with Gaussian kernel

minimum size is $\begin{cases} \lceil 6\sigma \rceil & \text{if } \lceil 6\sigma \rceil \text{ is odd} \\ \lceil 6\sigma \rceil + 1 & \text{otherwise} \end{cases}$



$\sigma = 7$
43x43



$\sigma = 7$
85x85



Difference

Smoothing with Gaussian kernel



Input image



$\sigma = 3.5$
21x21



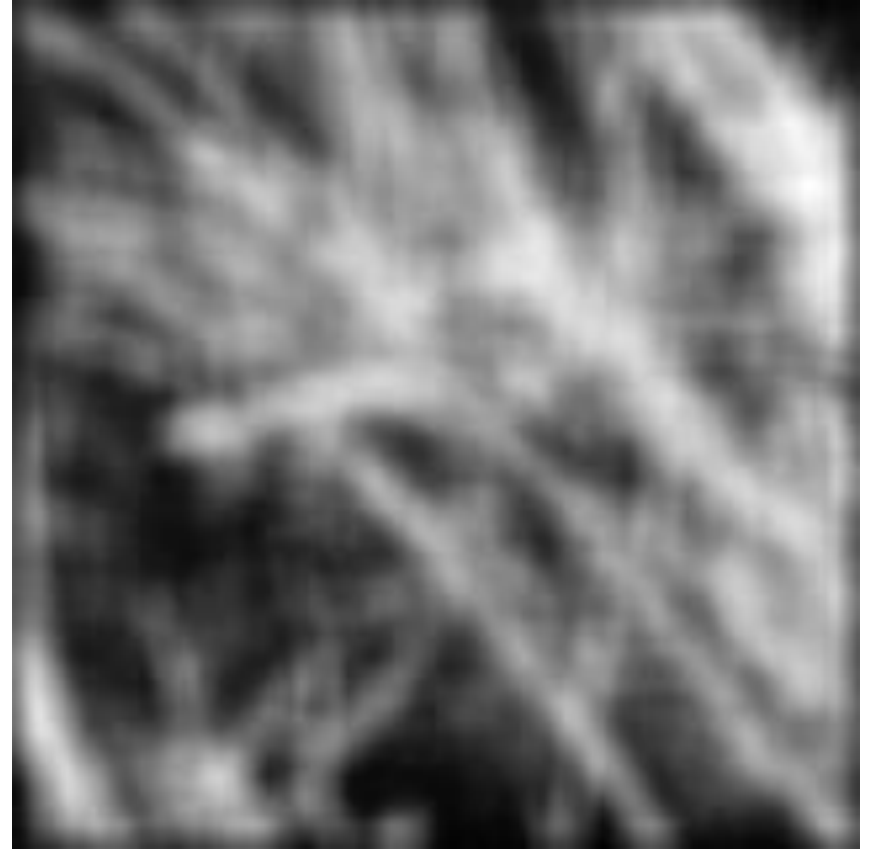
$\sigma = 7$
43x43

Smoothing with box kernel

Input image



Box kernel

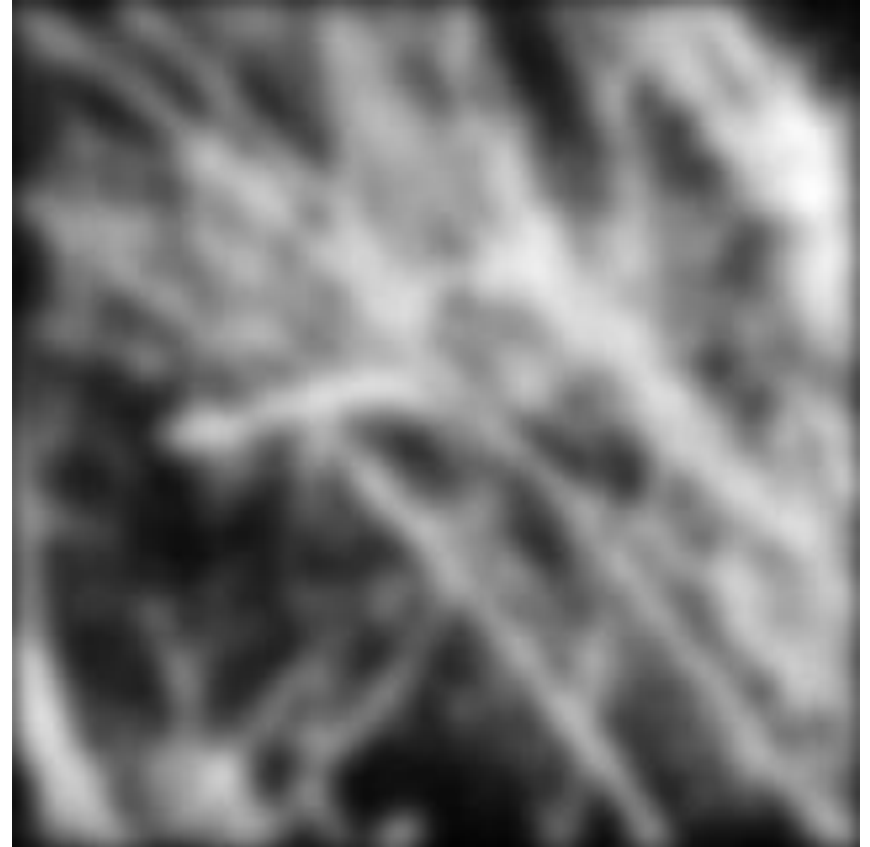


Smoothing with Gaussian kernel

Input image

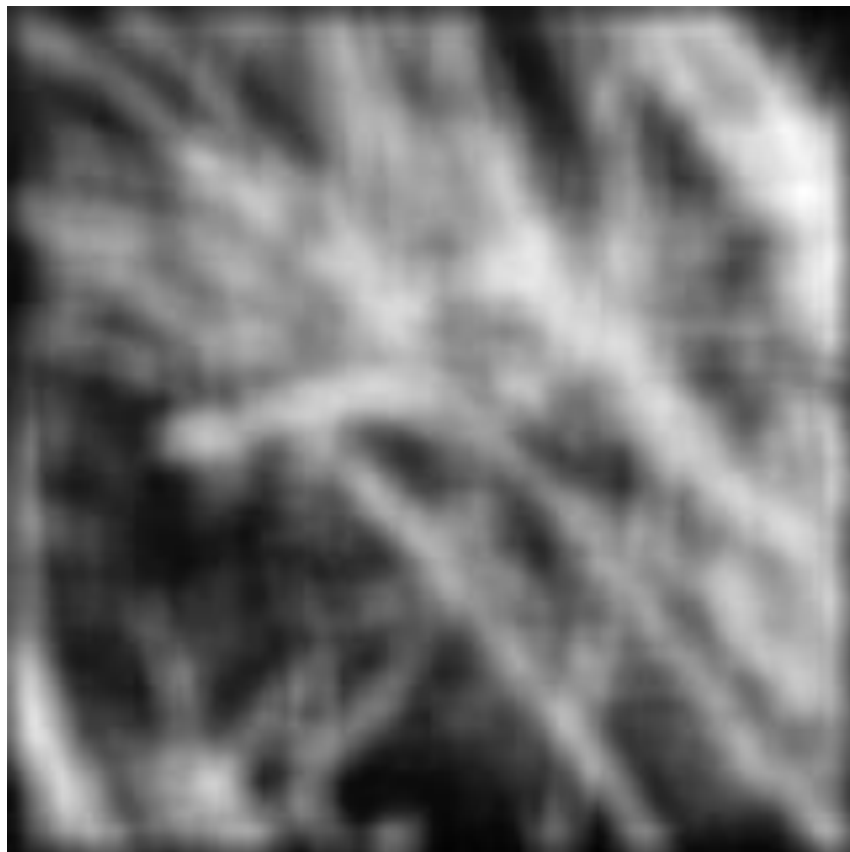


Gaussian kernel

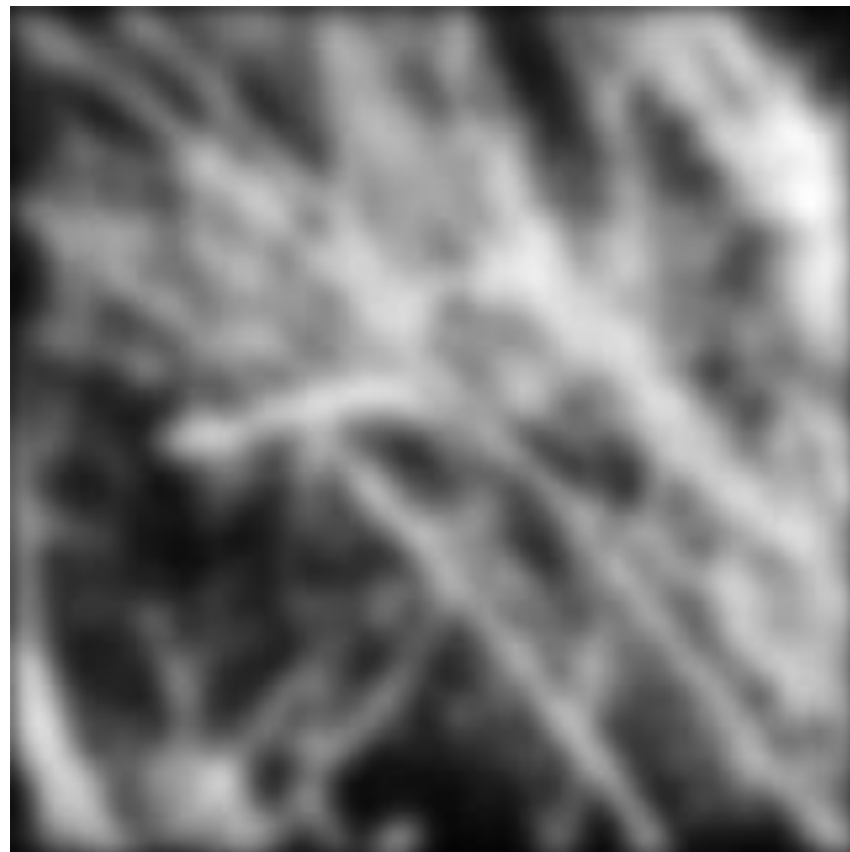


Smoothing

Box kernel



Gaussian kernel



Border padding

v	v	v	v	v	v	v	v	v
v	v	v	v	v	v	v	v	v
v	v	1	2	3	4	5	v	v
v	v	6	7	8	9	10	v	v
v	v	11	12	13	14	15	v	v
v	v	16	17	18	19	20	v	v
v	v	v	v	v	v	v	v	v

Zero padding
when $v = 0$

Constant padding

1	1	1	2	3	4	5	5	5
1	1	1	2	3	4	5	5	5
1	1	1	2	3	4	5	5	5
6	6	6	7	8	9	10	10	10
11	11	11	12	13	14	15	15	15
16	16	16	17	18	19	20	20	20
16	16	16	17	18	19	20	20	20

Replicate padding

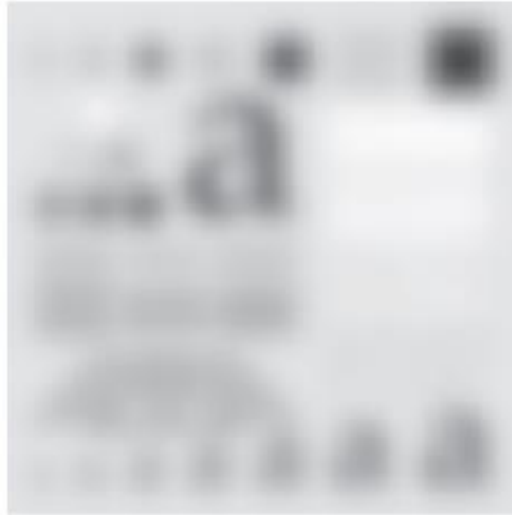
13	12	11	12	13	14	15	14	13
8	7	6	7	8	9	10	9	8
3	2	1	2	3	4	5	4	3
8	7	6	7	8	9	10	9	8
13	12	11	12	13	14	15	14	13
18	17	16	17	18	19	20	19	18
13	12	11	12	13	14	15	14	13

Mirror padding

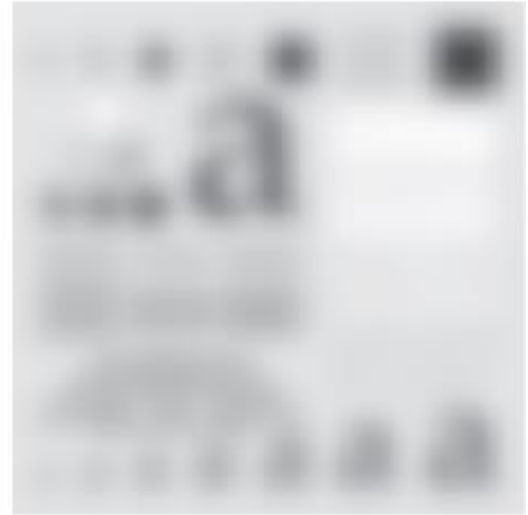
Border padding



Zero
padding



Mirror
padding



Replicate
padding

Bilateral filter

- An edge-preserving low pass filter

$$g(i, j) = \frac{\sum_{k,l} f(k, l) w(i, j, k, l)}{\sum_{k,l} w(i, j, k, l)}$$

– Bilateral weight function

$$w(i, j, k, l) = \exp \left(- \underbrace{\frac{(i - k)^2 + (j - l)^2}{2\sigma_d^2}}_{\text{Domain kernel}} - \underbrace{\frac{\|f(i, j) - f(k, l)\|^2}{2\sigma_r^2}}_{\text{Range kernel}} \right)$$

Domain kernel

Range kernel

Bilateral filter

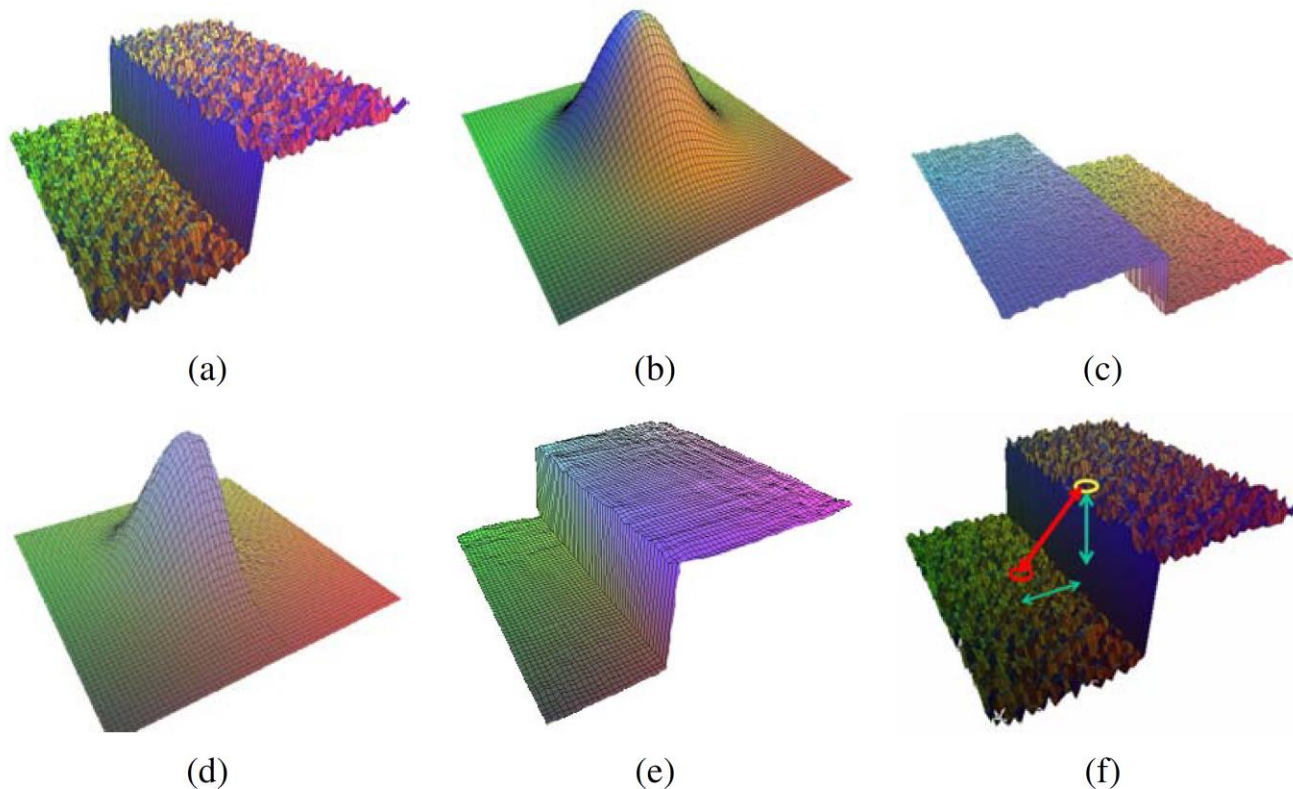
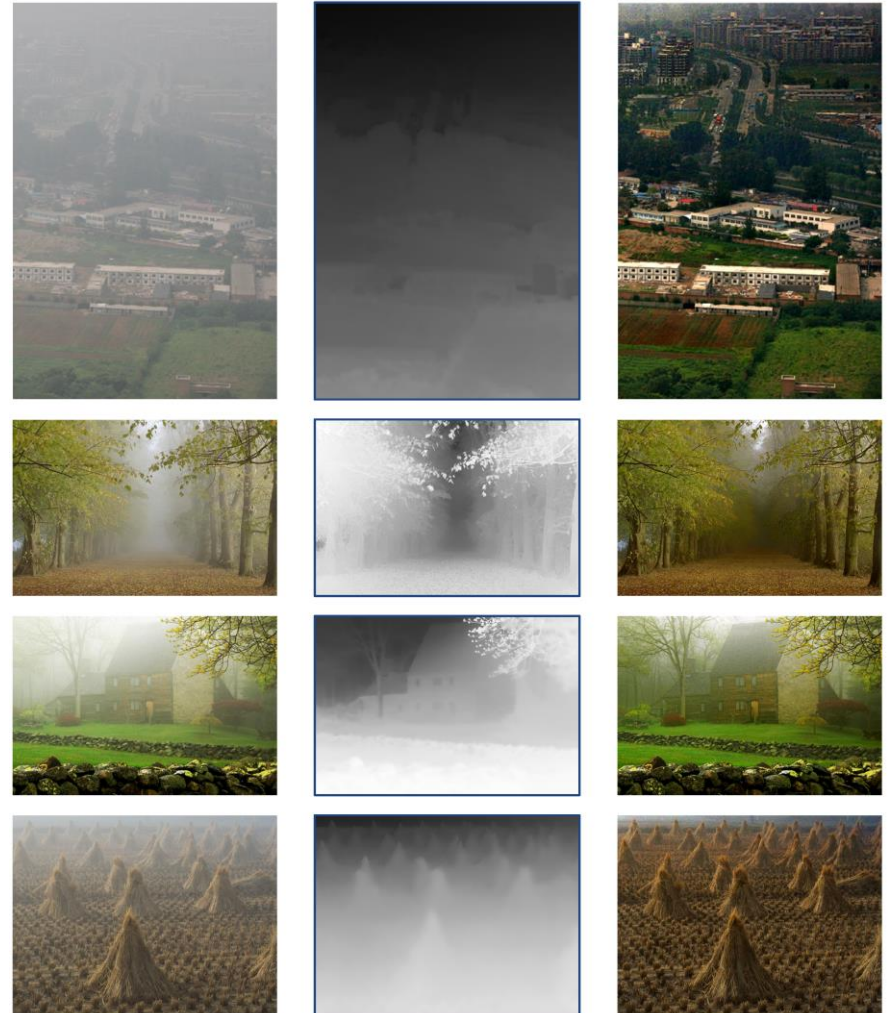


Figure 3.20 Bilateral filtering (Durand and Dorsey 2002) © 2002 ACM: (a) noisy step edge input; (b) domain filter (Gaussian); (c) range filter (similarity to center pixel value); (d) bilateral filter; (e) filtered step edge output; (f) 3D distance between pixels.

Dehazing

- “Single Image Haze Removal Using Dark Channel Prior”,
Kaiming He, Jian Sun, and Xiaoou Tang
 - Best paper, CVPR 2009



Dehazing

- “Single Image Dehazing via Multi-Scale Convolutional Neural Networks”, Wenqi Ren, Si Liu, Hua Zhang, Jinshan Pan, Xiaochun Cao, and Ming-Hsuan Yang

ECCV 2016



(a) Input

(b) He et al. CVPR2009

(c) Meng et al. ICCV2013

(d) Our



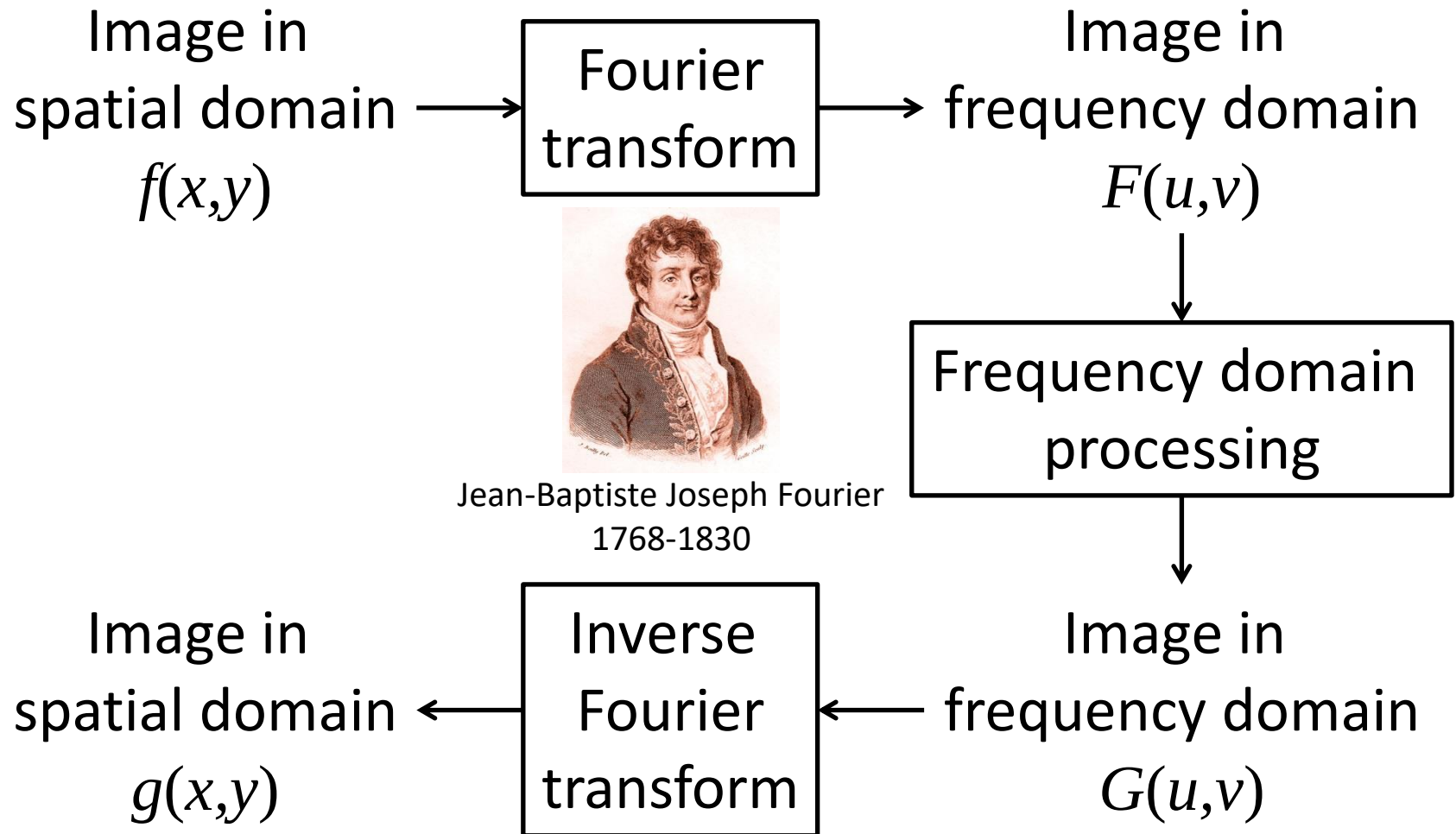
(a) Input

(b) He et al. CVPR2009

(c) Tang et al. CVPR 2014

(d) Our

Filtering in the frequency domain



2D discrete Fourier transform (DFT)

- (Forward) Fourier transform

Each of these are
an infinite,
periodic sequence
of copies

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)}$$

Units of cycles/sample

$$u = 0, 1, 2, \dots, M - 1 \text{ and } v = 0, 1, 2, \dots, N - 1$$

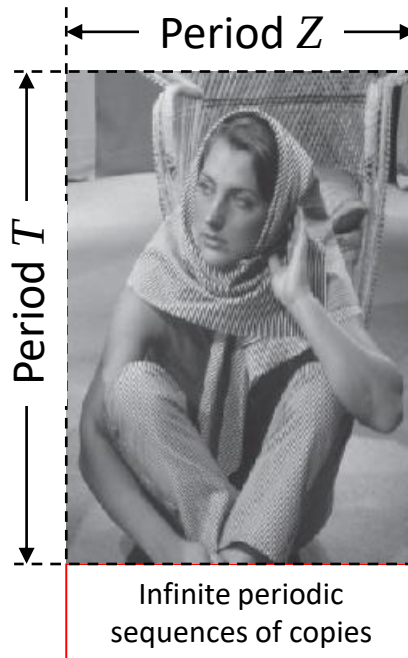
- Inverse Fourier transform

$$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)}$$

$$x = 0, 1, 2, \dots, M-1 \text{ and } y = 0, 1, 2, \dots, N-1$$

2D periodic functions

- Example periodic function



2D periodic functions

- Example periodic function

← Period Z →

Period T

Infinite periodic
sequences of copies

2D discrete Fourier transform (DFT)

- Fourier transform pairs

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)} \quad u = 0, 1, 2, \dots, M-1 \text{ and } v = 0, 1, 2, \dots, N-1$$

$$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)} \quad x = 0, 1, 2, \dots, M-1 \text{ and } y = 0, 1, 2, \dots, N-1$$

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)} \quad u = 0, 1, 2, \dots, M-1 \text{ and } v = 0, 1, 2, \dots, N-1$$

$$f(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)} \quad x = 0, 1, 2, \dots, M-1 \text{ and } y = 0, 1, 2, \dots, N-1$$

$$F(u, v) = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)} \quad u = 0, 1, 2, \dots, M-1 \text{ and } v = 0, 1, 2, \dots, N-1$$

$$f(x, y) = \frac{1}{\sqrt{MN}} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)} \quad x = 0, 1, 2, \dots, M-1 \text{ and } y = 0, 1, 2, \dots, N-1$$

2D discrete Fourier transform (DFT)

- Zero-frequency term (DC term)

Note zero-frequency term is proportional to $\bar{f}(x, y)$, the average value of $f(x, y)$

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)}$$

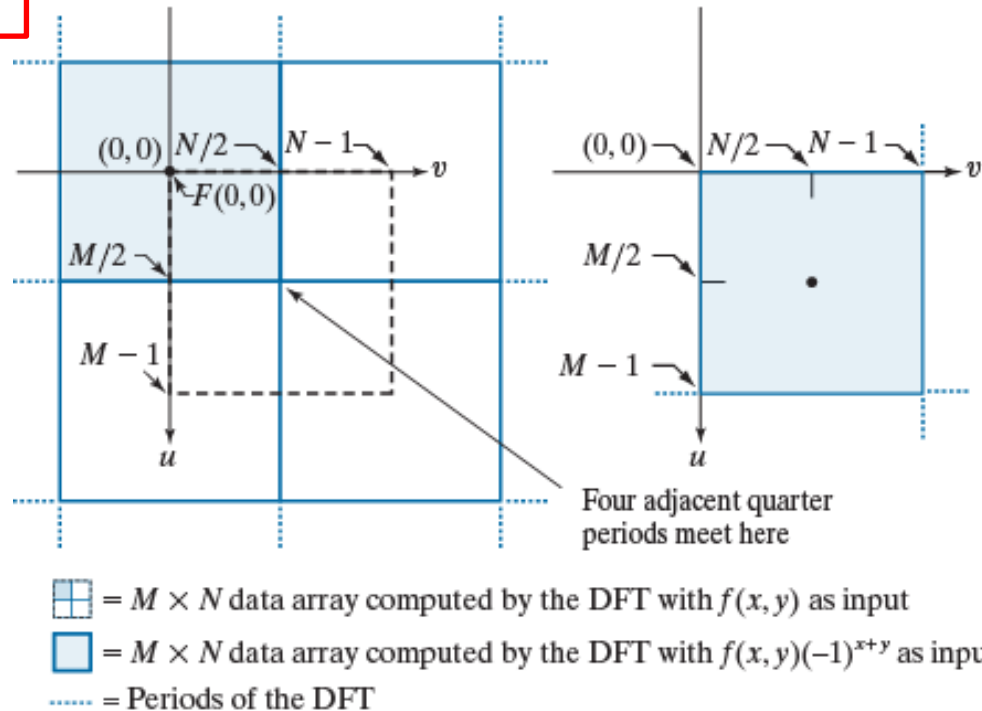
$$F(0, 0) = MN \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y)$$

$$F(0, 0) = MN \bar{f}(x, y)$$

$$\|F(0, 0)\| = MN \|\bar{f}(x, y)\|$$

Centering the DFT

This is an infinite, periodic sequence of copies

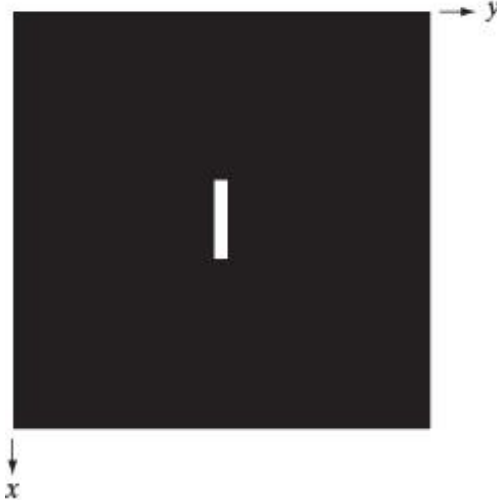


DFT showing an infinite number of periods. The area within the dashed rectangle is the data array, $F(u, v)$, obtained with Eq. (4-67) with an image $f(x, y)$ as the input. This array consists of four quarter periods. (d) Shifted array obtained by multiplying $f(x, y)$ by $(-1)^{x+y}$ before computing $F(u, v)$. The data now contains one complete, centered period, as in (b).

Centering the DFT

Each of these are an infinite, periodic sequence of copies

Original

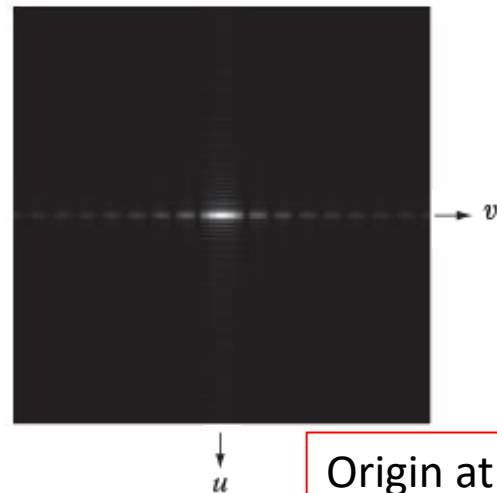


Origin at top left

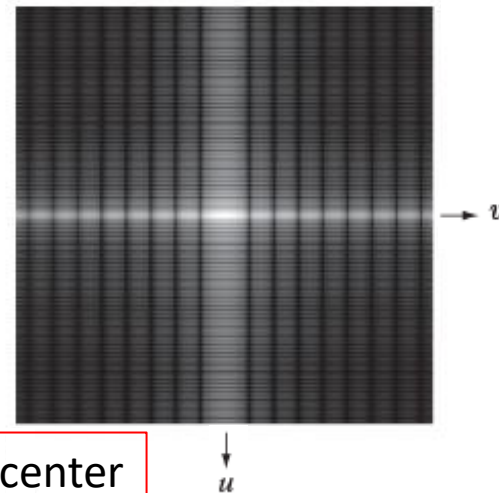


DFT magnitude
look at corners)

Shifted
DFT magnitude



Origin at center



Log of shifted
DFT magnitude

2D convolution

- 2D discrete convolution

$$f(x, y) \star h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) h(x-m, y-n)$$

$$x = 0, 1, 2, \dots, M-1 \text{ and } y = 0, 1, 2, \dots, N-1$$

- 2D convolution theorem

$$f(x, y) \star h(x, y) \iff F(u, v) H(u, v)$$

Key to filtering in the frequency domain

$$f(x, y) h(x, y) \iff F(u, v) \star H(u, v)$$

Circular convolution

- Because the DFT is an infinite, periodic sequence of copies, the convolution is **circular**

$$f(t) \star h(t) \iff H(\mu)F(\mu)$$

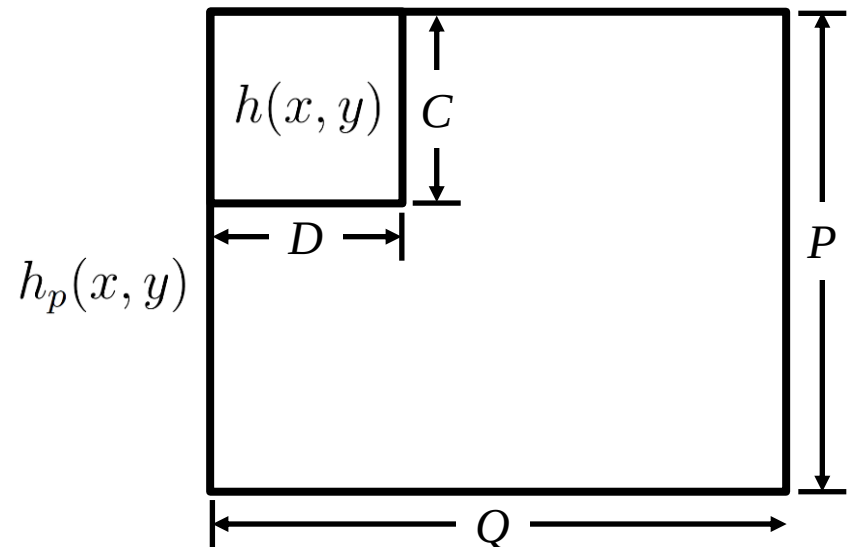
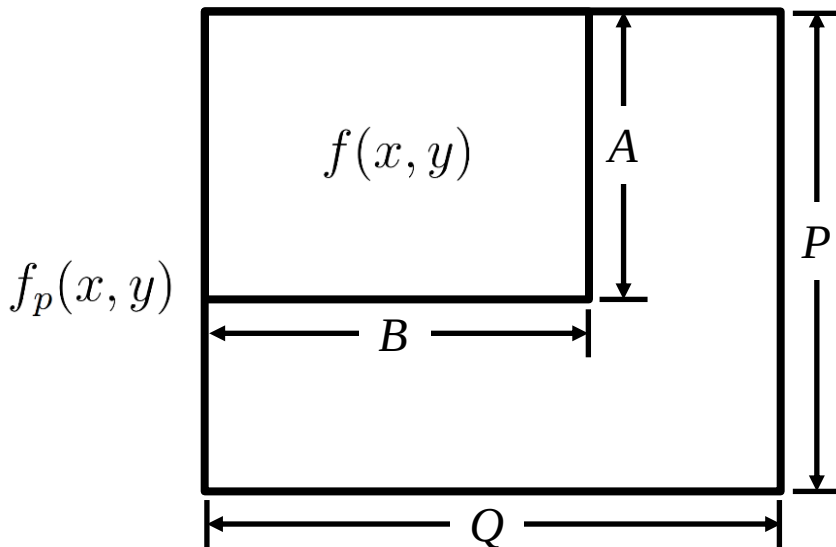
$$f(x, y) \star h(x, y) \iff F(u, v)H(u, v)$$

Key to filtering in the
frequency domain

Filtering using convolution theorem

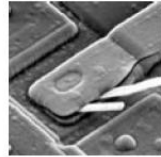
- To avoid wraparound error, both functions must be zero padded
- **2D** $f(x, y) \star h(x, y) \iff F(u, v)H(u, v)$

If $f(x, y)$ is $A \times B$ and $h(x, y)$ is $C \times D$, then both must be zero padded to $P \times Q$, where $P \geq A + C - 1$ and $Q \geq B + D - 1$

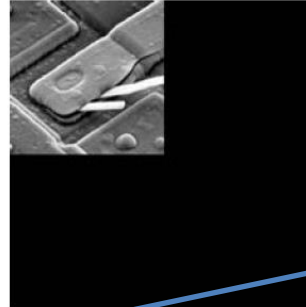


Filtering using convolution theorem

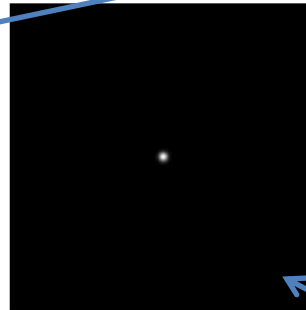
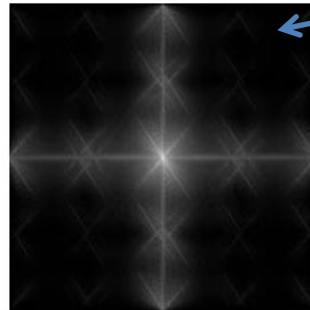
Input image



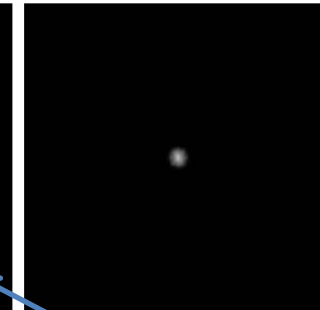
Zero padding



Fourier transform, then center

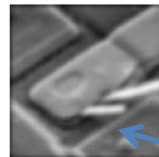
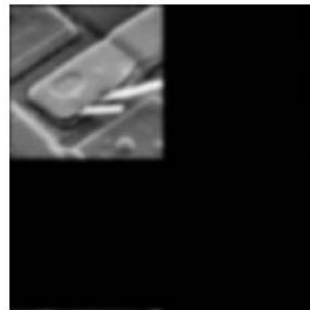


Product in frequency domain



Gaussian lowpass filter in frequency domain

Remove zero padding



Output image

Use real component only.

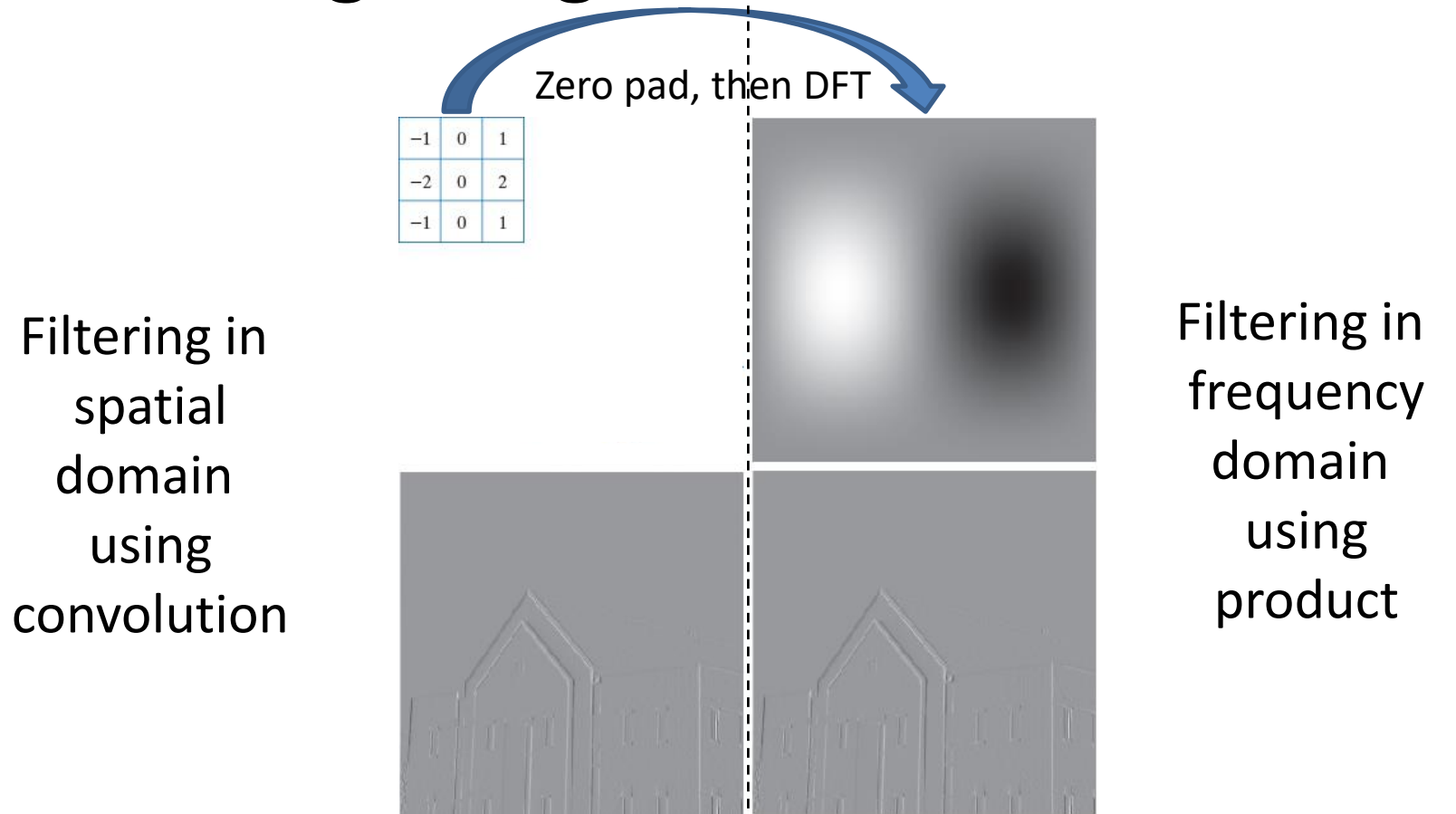
Mathematically, imaginary component will be 0, but due to floating point errors, it may be slightly nonzero.

Inverse center, then
inverse Fourier transform

Filtering in frequency domain using product with zero-padding

no wraparound error

Filtering using convolution theorem

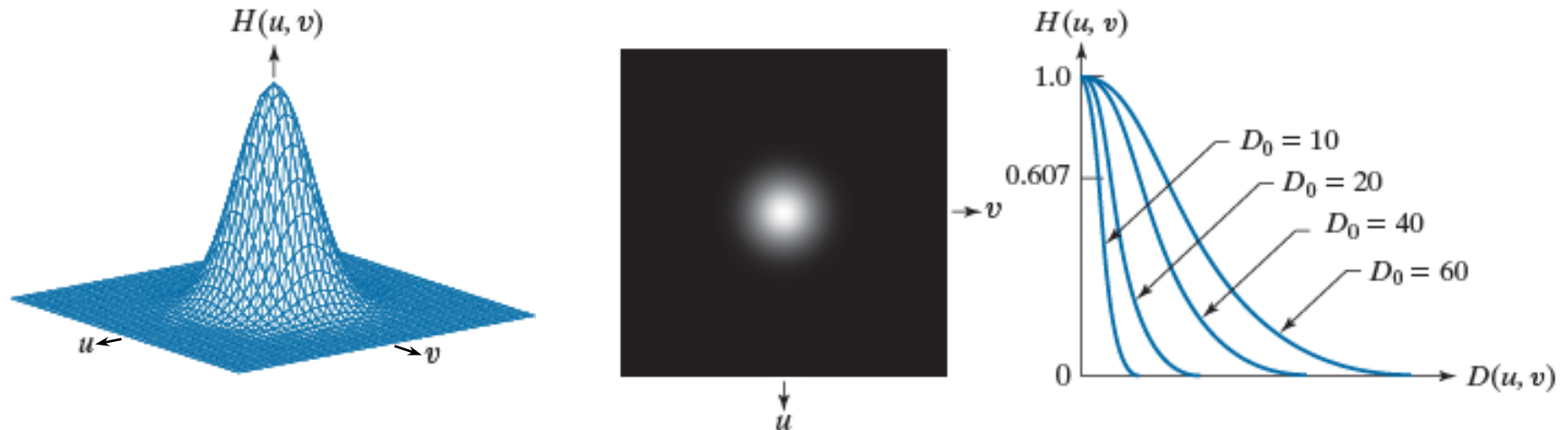


$$f(x, y) \star h(x, y) = \mathfrak{F}^{-1} \{F(u, v)H(u, v)\}$$

Identical results

Filtering in the frequency domain

- Gaussian lowpass filter (LPF)
 - Filter can be constructed in frequency domain directly



$$H(u, v) = e^{-D^2(u, v)/2\sigma^2}, \text{ where } D(u, v) \text{ is the distance from the center}$$

Letting $\sigma = D_0$, $H(u, v) = e^{-D^2(u, v)/2D_0^2}$

Filtering in the frequency domain

Gaussian LPF

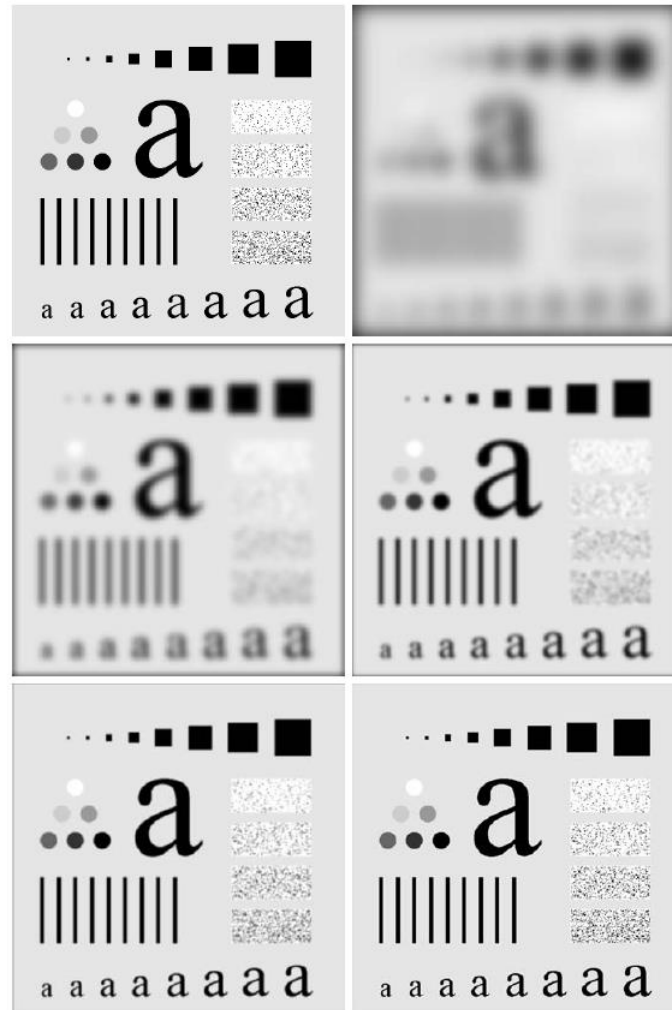
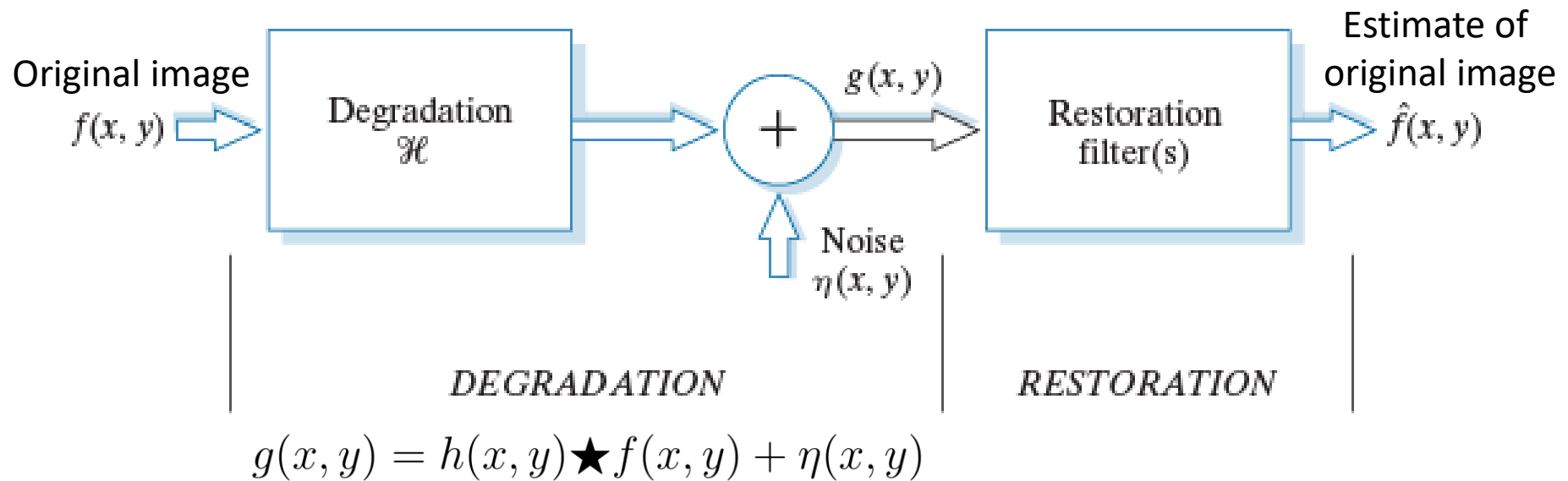


Image restoration

1. Remove noise
2. Estimate original image
 - Deconvolution

Model of image degradation, then restoration



Model of image degradation

- Spatial domain

$$g(x, y) = h(x, y) \star f(x, y) + \eta(x, y)$$

Degraded
image

Degradation
function

Original
image

Noise
image

- Frequency domain

$$G(u, v) = H(u, v)F(u, v) + N(u, v)$$

Degraded
image

Degradation
function

Original
image

Noise
image

Estimating the degradation function

- Methods
 - Observation
 - Experimentation
 - Mathematical modeling

Estimating the degradation function

- Observation

Manually process a subimage $g_s(x, y)$ of the degraded image $g(x, y)$ and let the result be our estimate of the original subimage $\hat{f}_s(x, y)$

$$g_s(x, y) = h_s(x, y) \star \hat{f}_s(x, y), \text{ solve for } h_s(x, y)$$

This is called blind deconvolution because the true degradation function is not known.
Solve in the frequency domain

$$G_s(u, v) = H_s(u, v) \hat{F}_s(u, v)$$
$$H_s(u, v) = \frac{G_s(u, v)}{\hat{F}_s(u, v)}$$

then construct $H(u, v)$ for whole image with the same basic shape as $H_s(u, v)$

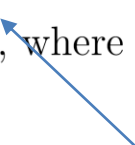
Estimating the degradation function

- Experimentation

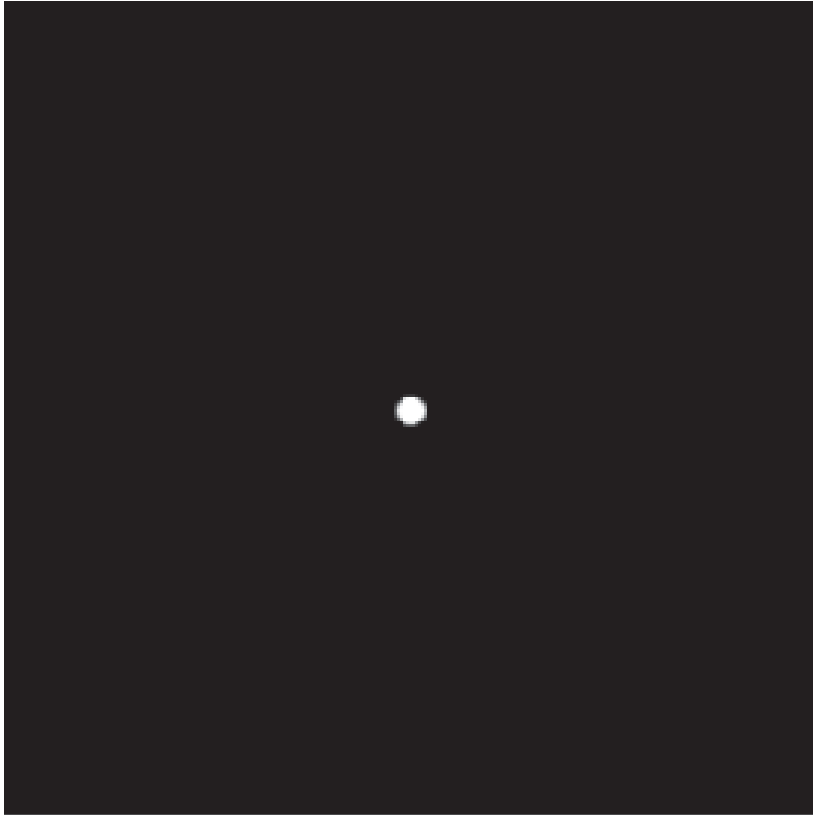
If the imaging system is available, then obtain the impulse response of the degradation by imaging an impulse (i.e., a small dot of light, as bright as possible)

$$H(u, v) = \frac{G(u, v)}{A}, \text{ where } A \text{ is a constant describing the strength of the impulse}$$

DFT of acquired, degraded image

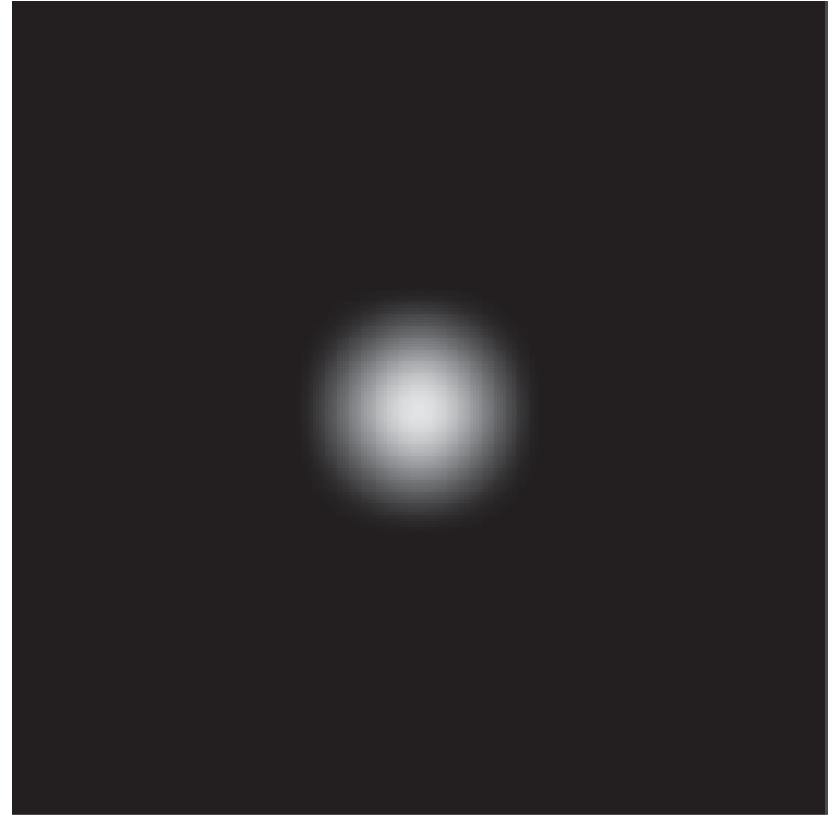


Estimation of degradation function by experimentation



Impulse of light

$$f(x, y)$$



Imaged (degraded) impulse

$$g(x, y)$$

Estimating the degradation function

- Mathematical modeling

Model $H(u, v)$ known parameters

Estimation of degradation function by mathematical modeling

$f(x, y)$



Severe turbulence
 $g(x, y)$



Atmospheric
turbulence
model

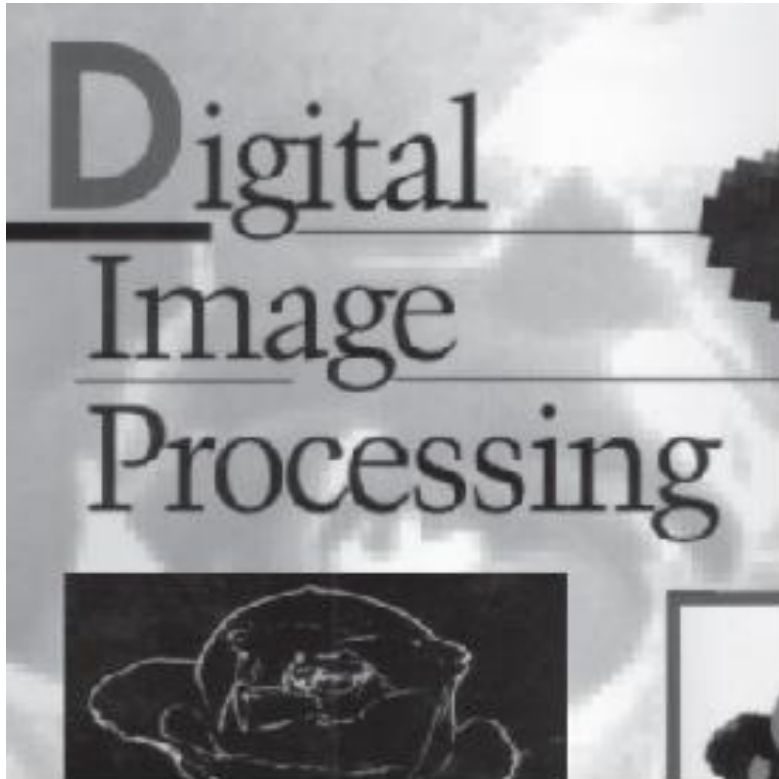
Mild turbulence
 $g(x, y)$



Low turbulence
 $g(x, y)$



Estimation of degradation function by mathematical modeling



$f(x, y)$

Motion blur model



$g(x, y)$

Image restoration

- Inverse filtering

$$G(u, v) = H(u, v)\hat{F}(u, v)$$

$$\hat{F}(u, v) = \frac{G(u, v)}{H(u, v)}$$

- Mitigate divide by zero

- Apply window function to $g(x, y)$
- Ideal lowpass filter $H(u, v)$
- Threshold $H(u, v)$
$$\hat{F}(u, v) = \begin{cases} \frac{G(u, v)}{H(u, v)} & \text{if } |H(u, v)| \geq t \\ 0 & \text{otherwise} \end{cases}$$

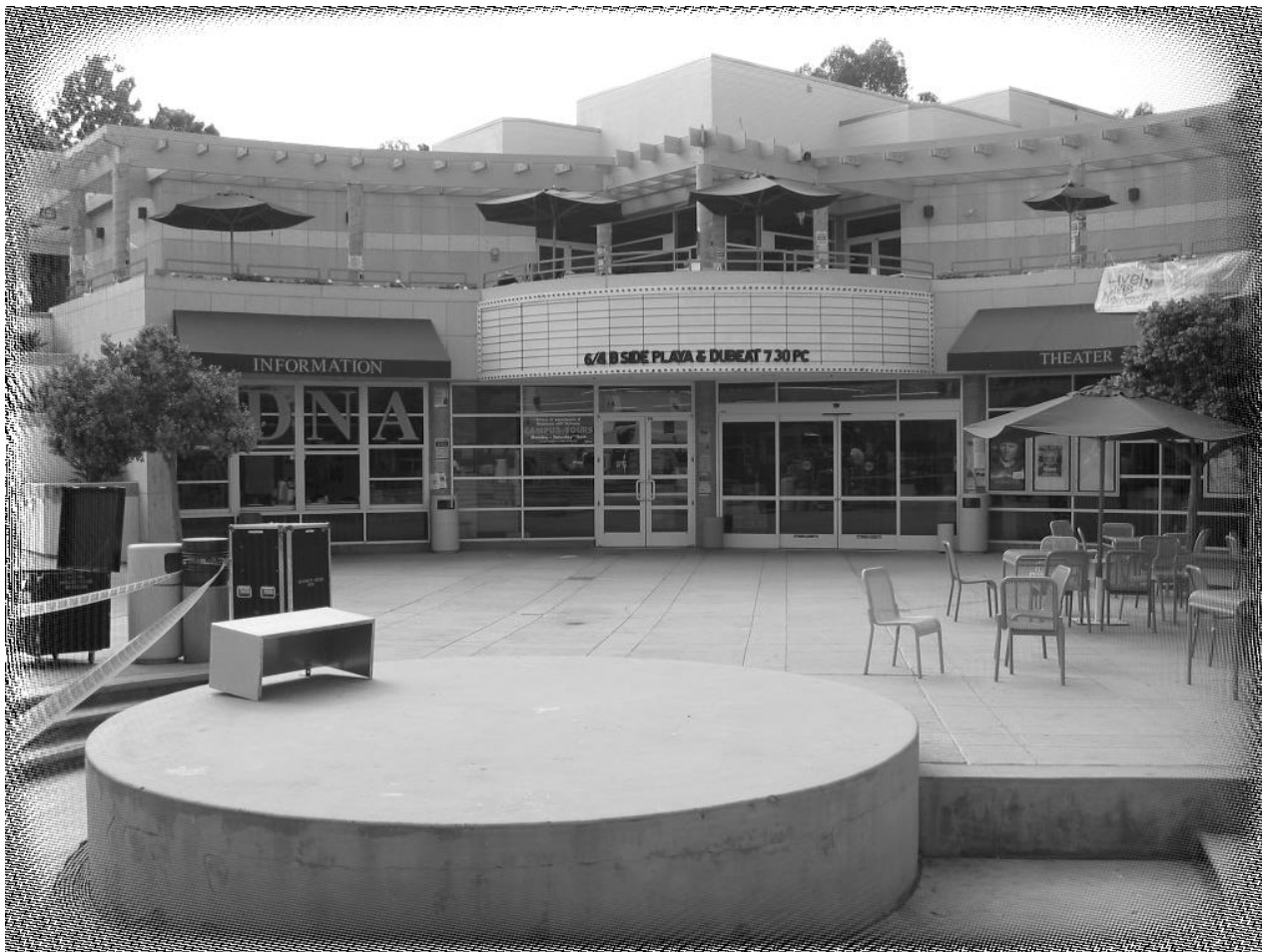
Image restoration, inverse filtering



$$g(x, y)$$

Image restoration, inverse filtering

Hann(ing)
window
applied to
 $g(x,y)$

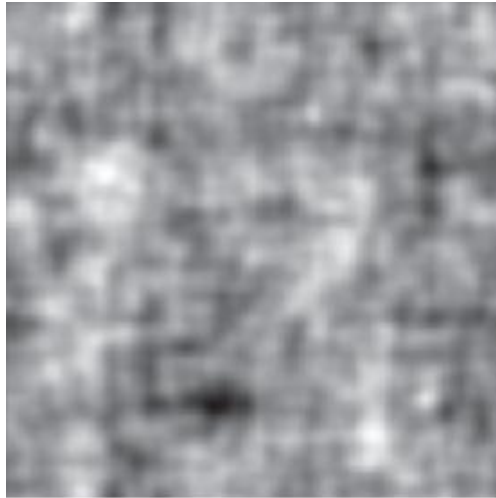


$$\hat{f}(x,y)$$

Image restoration, inverse filtering

Full (i.e., no
lowpass
filter)

$$\hat{f}(x, y)$$



Lowpass
filter
limited to
radius of 40

$$\hat{f}(x, y)$$



Lowpass
filter
limited to
radius of 70

$$\hat{f}(x, y)$$



Lowpass
filter
limited to
radius of 85

$$\hat{f}(x, y)$$

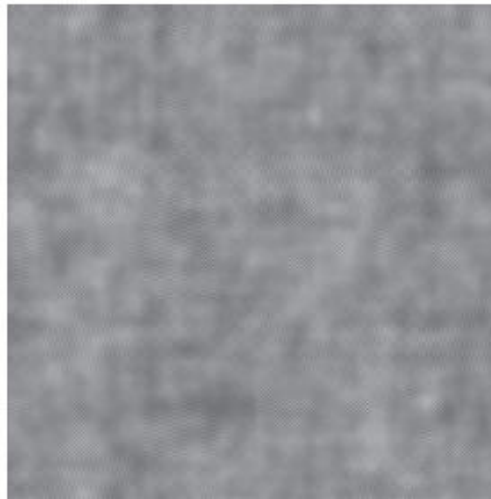
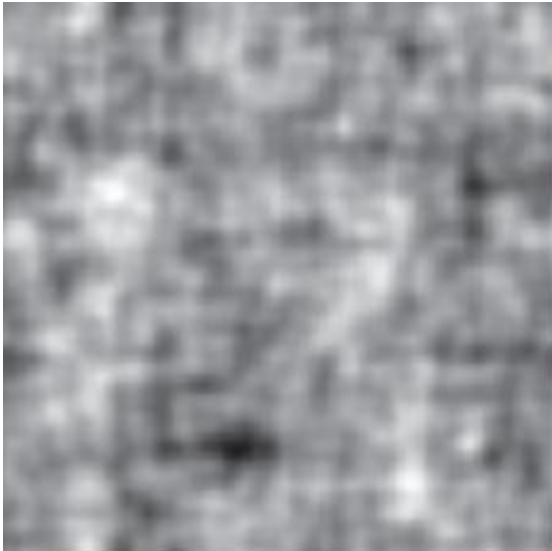


Image restoration, Wiener filtering

Inverse filtering



Full

$$\hat{f}(x, y)$$



Lowpass filter
radially limited
 $\hat{f}(x, y)$

CSE 273, Winter 2024

Wiener filtering



$\hat{f}(x, y)$

$$e^2 = E\{(f - \hat{f})^2\}$$

Minimum mean
square error filter

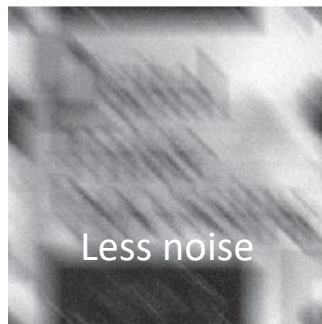
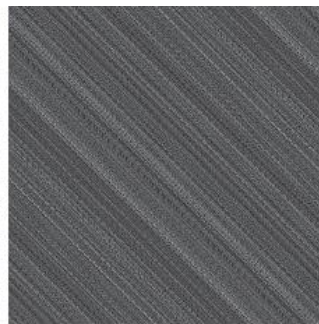
Image restoration, constrained least squares filtering

Degraded
image

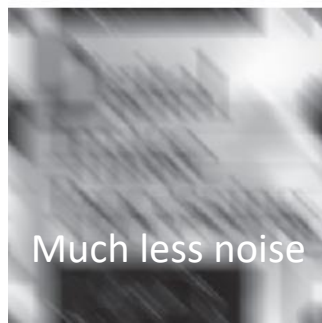
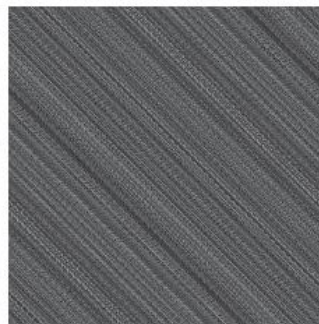
Inverse
filtering

Wiener
filtering

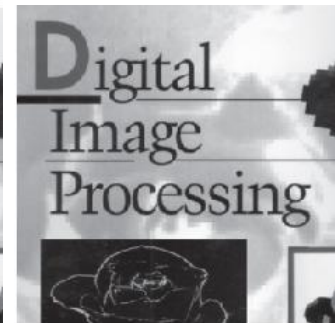
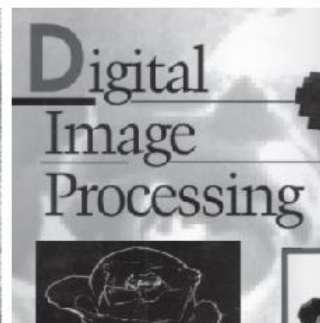
Constrained least
squares filtering



Less noise



Much less noise



Next lecture

- Set of standard dynamic range (SDR) images to a high dynamic range (HDR) image